Application of Learning Algorithms to Traffic Management in Integrated Services Networks

Jason Lee Hall

Centre for Telecommunication Networks

School of Engineering

University of Durham

December 1998

Submitted for the degree of Doctor of Philosophy (Ph.D)

of the University of Durham.

Abstract

Future integrated services networks are expected to support a wide range of applications which have diverse traffic characteristics and quality of service requirements. The fundamental problem with network control are the conflicting aims of maximising utilisation and satisfying quality of service requirements, whilst also using a simple set of traffic descriptors. Services based on measurements and predictions aim to increase utilisation whilst working on the premise of a simple traffic descriptor, however there is also a risk that quality of service requirements will not be satisfied. The learning and adaptation properties of artificial intelligence techniques appear well suited to this domain and this thesis studies the potential of applying them to network control.

We investigate the application of neural networks to perform traffic prediction by learning the relationship between past and future traffic variations. Contrary to popular belief we conclude that generally this is only possible with artificial traffic generated using mathematical models or traffic which exhibits periodic properties, and that accurate prediction of real network traffic is generally impossible. The adaptive features of learning automata are utilised in a new dynamic scheduling mechanism for a packet multiplexer based on measurements. By minimising the resources allocated to each traffic flow the scheme is able to satisfy a variety of quality of service requirements whilst also maximising potential utilisation. Simulation results demonstrate that the automata are able to converge to optimal probabilities and are able to perform well over a much wider range of traffic conditions compared to traditional scheduling mechanisms. A novel admission control scheme is described for the adaptive scheduler in which a neural network learns the required probability based on a token bucket traffic descriptor. The mathematical form of this mapping is unknown and the neural network learns the function based on examples obtained through simulation.

Declaration

I hereby declare that this thesis is a record of work undertaken by myself, that has not been the subject of any previous application for a degree, and all sources of reference have been duly acknowledged.

Acknowledgements

I wish to express my thanks to my supervisor, Professor Phil Mars, for his guidance, encouragement and friendship over the past three years.

I gratefully acknowledge EPSRC and Fujitsu Microelectronics Ltd for financial support and Jim Stone and Graeme Smith of Fujitsu for their support and for making my stays with them run smoothly.

Thanks also to my colleagues in the lab who have helped me out over the years and who have also made for a pleasant working environment, in particular thanks to Jonathan and Mark.

My time in Durham would have been far less enjoyable if it wasn't for my friends outside of work. I appreciate them all and in particular thanks must go to Ste, Dave and Lee for contributing to some enjoyable and interesting evenings!

Finally I must thank my parents for supporting me throughout my time in Durham.

Table of Contents

List of Figures

List of Tables

List of Abbreviations

ABR	Available Bit Rate
AI	Artificial Intelligence
ATM	Asynchronous Transfer Mode
BT	Burst Tolerance
CBR	Constant Bit Rate
CDVT	Cell Delay Variation Tolerance
CLT	Central Limit Theorem
Delay-EDD	Delay Earliest Due Date
EBCN	Explicit Backward Congestion Notification
EDF	Earliest Deadline First
EFCN	Explicit Forward Congestion Notification
F-ARIMA	Fractional Autoregressive Integrated Moving Average
FBM	Fractional Brownian Motion
FFT	Fast Fourier Transform
FGN	Fractional Gaussian Noise
FIFO	First In First Out
FIR	Finite Impulse Response
FQ	Fair Queueing
FSRR	Framed Selection with Round Robin
FSRRBE	Framed Selection with Round Robin excluding Best Effort
GA	Genetic Algorithm
GPS	Generalised Processor Sharing
HOL	Head Of Line
HOL-PJ	Head Of Line Priority Jumps
HRR	Hierarchical Round Robin
IETF	Internet Engineering Task Force
IFS	Iterative Framed Selection
ISS	Iterative Stochastic Selection
ITU	International Telecommunications Union

Jitter-EDD	Jitter Earliest Due Date
LAN	Local Area Network
LCFD	Last Come First Drop
MLP	Multi Layer Perceptron
MMBP	Markov Modulated Bernoulli Process
MMDP	Markov Modulated Deterministic Process
MMPP	Markov Modulated Poisson Process
MSE	Mean Squared Error
MUST	Maximum Utility Scheduling for Transmission
NN	Neural Network
NRT-VBR	Non Real-Time Variable Bit Rate
PDF	Probability Density Function
QoS	Quality of Service
RCSP	Rate Controlled Static Priority
RMD	Random Mid-point Displacement
RPQ	Rotating Priority Queues
RR	Round Robin
RTT	Round Trip Time
RT-VBR	Real-Time Variable Bit Rate
SCFQ	Self Clocked Fair Queueing
SP	Static Priority
SSRR	Stochastic Selection with Round Robin
SSRRBE	Stochastic Selection with Round Robin excluding Best Effort
TCP	Transmission Control Protocol
TDM	Time Division Multiplexing
TDNN	Time Delay Neural Network
UBR	Unspecified Bit Rate
VBR	Variable Bit Rate
WAN	Wide Area Network
WF^2Q	Worst Case Fair Weighted Fair Queueing
WFQ	Weighted Fair Queueing
WRR	Weighted Round Robin

Chapter 1

Introduction

Unlike current communication networks which are designed to offer a special type of service, integrated services networks will offer multiple services to support applications that include voice, video, data and many others. This integration of services onto a single network will offer a number of advantages, such as improved statistical multiplexing, economies of scale and widespread access. However, the integration also presents new challenges for network designers in terms of traffic management and congestion control. In this chapter we begin by presenting some background and motivation for packet switched integrated services networks. Next we describe various key elements essential for an integrated services architecture and finally we present an outline of the thesis.

1.1 Background

Traditional communication networks offer a single type of service that supports one application. For example, the telephone network has been designed to support interactive voice which requires a low delay, low bandwidth, two-way, jitter-free service. By contrast, the cable television network has been designed to support broadcasting of analogue video which requires a high bandwidth, one-way, jitter-free service. These specific goals have allowed network design to be optimised for each particular type of service.

Integrated services networks will have to support various existing applications together with many new and radically different applications, often with diverse traffic characteristics and performance objectives. Because of this diversity, together with the uncertainty of future applications, a network should be flexible in its ability to support many different applications. Although traditional packet switching data networks are highly flexible and can support a wide range of data applications, they are unable to provide any performance guarantees. Circuit switched voice networks can provide excellent guarantees, however they are inflexible and can only provide guarantees to a limited number of applications. Therefore the design of integrated

services networks is a challenging problem, since they need to be both flexible and provide guaranteed service.

It has been widely accepted that future integrated services networks will use packet switching technology. The reasons behind this are flexibility and an ability to exploit statistical multiplexing. In a packet network, sources statistically share network resources and potentially there are an unlimited number of users that can use the network at any one time. This is because in a networking environment where traffic sources are bursty, it is unlikely that all sources will transmit at their peak rates simultaneously. Therefore allocating each flow a bandwidth less than its peak rate gives rise to the statistical sharing of network resources and can lead to higher network utilisation compared to circuit-switched networks that work on peak rate allocation. However, statistical multiplexing introduces the problem of congestion since there is a possibility that instantaneous demand will be greater than the available resources, resulting in increased delays and packet loss. Although advancing technology will lower the cost of memory and increase link capacities and processor speeds, the problem of congestion is unlikely to go away [1]. Therefore an integrated services network architecture with appropriate congestion control procedures is required such that the fundamental goal of network design is achieved, namely to satisfy the requirements of the user [2]. A further subsidiary goal of network design is to maximise the utilisation of the network such that revenue can be maximised. To achieve these goals it is necessary to have a thorough understanding of the characteristics and requirements of the applications to be supported.

1.2 User Applications

The introduction of many new applications leads some people to believe that we cannot design future networks by looking at the current situation [3]. However, current applications will continue to exist and they represent a wide range of features and requirements upon which we can base future networks.

Applications can be divided into approximately five classes, these are conversational, messaging, distribution, retrieval and machine-to-machine applications [3]. Conversational or interactive applications consist of a person at each end of the connection. Messaging applications involve a person talking to a machine and distribution applications consist of a machine sending to people or machines who listen passively. A retrieval application is one in which a person causes a machine to transfer information and machine-to-machine applications involve communication

between computers. A list of application classes and corresponding examples is given in Table 1.1 [3].

The degree to which application performance depends upon delay varies widely and from the list of applications in Table 1.1 we can classify them as either real-time or non-real-time.

Application Class	Example Applications
Interactive video	Video conferencing
Interactive audio	Telephone
Interactive text/data	Credit card verification
Interactive image	Multimedia conferencing
Video messaging	Multimedia e-mail
Audio messaging	Voice mail
Text/data messaging	Electronic mail
Image messaging	High resolution fax
Video distribution	Television
Audio distribution	Radio
Text distribution	Teletext
Image distribution	Weather satellite pictures
Video retrieval	Video on demand
Audio retrieval	Audio library
Text/data retrieval	File transfer
Image retrieval	Library browsing
Remote procedure call	Distributed simulation
Distributed file service	

Table 1.1:List of application classes and examples

Real-time applications require the delivery of packets by a certain time, if the data has not arrived by then it is essentially worthless. The interactive and distribution classes fall into this category, since they involve people observing a continuous flow of visual or auditory information and excessive delay, delay variation or loss has a significant impact on perceived quality. Alternatively, non-real-time or elastic applications will always wait for the data to arrive and the messaging and retrieval classes fall into this category. Such applications may still have different sensitivities to delay, for example messaging applications are very tolerant of delay whereas with

retrieval applications the user may be somewhat sensitive to long delays.

1.2.1 Real-Time Applications

The most common type of real-time application is likely to be a playback application [4]. In a playback application, the source packetises a signal and transmits it across the network. The network inevitably introduces some variation in the delay of each packet, this variation is known as jitter. On arrival at its destination, the receiver de-packetises the data and attempts to playback the signal, this is achieved by buffering the incoming data in order to remove the jitter and replaying the signal at some pre-determined playback point. Data that arrives before the playback point can be used to reconstruct the signal, however data arriving after the playback point is worthless. Therefore in order to choose a reasonable offset for the playback point, an application needs some knowledge of the maximum delay its packets will experience.

The performance of a playback application can be measured in two ways : latency and fidelity [5]. The latency is the delay between when the signal is generated at the source and when it is played back at the receiver. Some applications are more sensitive to latency than others, for example interactive applications are more sensitive than other playback applications such as transmitting a movie, since in the latter case the users watch passively. Fidelity is a measure of the quality of the signal reproduced at the receiver. The playback signal is incomplete whenever packets are late or lost due to buffer overflow within the network and distorted whenever the offset is varied. As with latency, applications differ in their sensitivity to fidelity, for example a video conference allowing one surgeon to remotely assist another during an operation requires a much higher fidelity than a video conference based family re-union, since in the latter case the users can easily compensate for slight glitches in transmission [4]. These differences in sensitivity allow playback (and other real-time) applications to be categorised as either tolerant or intolerant [5].

Intolerant applications must have a fixed playback point since any variation of this offset will introduce distortions in the playback signal. Furthermore, this fixed offset must bound the maximum end-to-end delay in order to avoid late packets. By contrast, tolerant applications can have a reduced offset since they can tolerate the lower quality in the recreated signal when some packets arrive after the playback point. Moreover, the offset may be adjusted depending on the delays experienced by incoming packets, since these applications can tolerate the resulting distortion in the signal. Instead of reducing the offset, a fixed offset may allow for a greater admitted load onto the network which again runs the risk of packets arriving after the playback

point.

1.2.2 Non-Real-Time Applications

Although some applications are called non-real-time, it is wrong to assume that they are not sensitive to delay. Typically, the application will use the arriving data immediately and will always wait for incoming data rather proceed without it, thus significantly increasing the delay of a packet may harm performance. Generally, the performance of these applications will depend more on the average delay rather than on the tail of the distribution, which is important for real-time applications. The delay requirements of such applications are varied, for example electronic mail has less of a requirement than file transfer. As with real-time applications it is also possible to identify loss tolerant and loss intolerant applications. For example, a voice mail message may be more tolerant to loss than an image message since the receiver may be able to compensate for the slight interruptions in the reconstructed voice, however the fact that these are non-real time applications may make retransmission a viable option.

1.3 Integrated Services Network Architecture

The most difficult technical problem that blocks the path towards an integrated services network is that of supporting real-time applications, since these require performance guarantees. Currently there are two widely accepted architectures for packet-switched integrated services networks, the International Telecommunications Union (ITU) recommendation is Asynchronous Transfer Mode (ATM) and the Internet Engineering Task Force (IETF) recommendation is an integrated services extension to the current Internet called IntServ. The main difference between the two is that ATM uses small fixed size packets, called cells, whereas IntServ assumes variable length packets. Both proposals however are based on the common paradigm that before communication starts, the client must specify its traffic characteristics and desired performance objectives. Based on this, an admission control policy decides whether or not to accept the connection. If the network accepts the request, it makes some kind of commitment that the performance requirements will be met providing the client obeys its traffic specification. Clark et al [4] identify several key components in an integrated services architecture, namely the service interface, the service commitments and congestion control. We now consider briefly each of these components.

1.3.1 Service Interface

The service interface governs the set of parameters that describe the traffic flow requesting access to the network, these must include the quality of service (QoS) objectives and an estimate of the traffic characteristics. The QoS parameters are likely to depend upon the application, however typical parameters will include loss rate, mean delay, maximum delay and maximum delay variation. The parameters that will describe the traffic characteristics are less well defined and the conflicting requirements of describing the traffic characteristics using minimal parameters yet providing maximum information, ensures that optimal traffic characterisation remains an open issue.

1.3.2 Service Commitments

The service commitment determines the nature of the guarantee made by the network when it chooses to accept a connection. Service commitments can be divided into two categories, the first is a QoS commitment to individual flows whereas the second is a commitment to resource sharing in which the network makes assurances that the resource in question will be shared according to some protocol. Since the primary goal of network design is to satisfy the requirements of the user, we argue that the QoS commitments are the most important.

The current Internet does not make any QoS commitments and simply treats all packets equivalently. A key point regarding this is that the current Internet only requires a trivial admission controller that admits all connections, this may result in severe congestion and massive delays. This type of service is known as best-effort. In [6], two other types of service commitment are identified as being necessary for integrated services networks, namely deterministic and statistical. However, there has been recent widespread support for services based on measurements [4][5] which act to increase utilisation at the risk of packets violating QoS requirements.

- Deterministic Service

  A deterministic service commitment provides an absolute bound on the performance of all packets within a session. Intolerant applications require this type of service and a typical deterministic service commitment is one which guarantees that no packets will suffer an end-to-end delay greater than S. These commitments should be honoured regardless of the congestion in the network, therefore such a service often results in low utilisation. An example of an intolerant application requiring a deterministic service

commitment is nuclear power plant control, since violations of such guarantees may result in catastrophic consequences.

- Statistical Service

  A statistical service commitment provides a statistical bound on the performance of all packets within a session. Such a service is able to exploit statistical multiplexing, since the occasional performance violation allows for greater utilisation which will result in a presumed lower cost for the service, therefore tolerant applications would benefit from using this type of service. A typical statistical commitment is one which guarantees that no more than $\varepsilon\%$ of packets will suffer an end-to-end delay greater than S. However, in practise, the precise definition of a statistical guarantee is rather difficult. For example over what time interval should the statistics be measured? One would expect that 100 consecutive late packets out of 100000 would have more of an effect than one late packet out of every 1000. For this reason, interval-based statistical QoS guarantees have been proposed [7], but the question of the appropriate time interval remains an open question and is application specific. Several multimedia applications such as audio and video can tolerate some degree of late or lost packets and are therefore well suited to a statistical service.

- Measurement Based Service

  A measurement based service commitment provides 'fairly' reliable deterministic or statistical bounds. Deterministic and statistical services rely heavily on the use of a priori client specified parameters when determining the resource requirements for a flow. Consequently, these services may not be well suited to applications which cannot accurately estimate their traffic parameters, such as live video, or applications with rate variations over multiple time scales which are not adequately characterised by standard models. In these situations conservative estimations must be made and may result in poor network utilisation. In contrast, measurement based services rely on measured values of traffic parameters rather than a priori client specified guesses, therefore the effects of inaccurate traffic specification can be alleviated and high utilisation can be achieved. Due to the inherent dynamic nature associated with network traffic it is impossible to make firm guarantees, however by applying suitable control procedures it is possible to provide fairly reliable or soft performance guarantees. With this type of service the network is effectively gambling that past network behaviour is a good indicator of future network behaviour.

Based on these commitments, service models for ATM and IntServ have been proposed, however it must be noted that these models are by no means fixed and additional services may be added in the future.

1.3.2.1 ATM Service Model

As previously mentioned, the single most important difference between applications is whether they are real-time or non-real time. The two most basic service categories to support these in ATM are the constant bit rate (CBR) and unspecified bit rate (UBR). CBR reserves a fixed bandwidth to each connection and provides a simple reliable service which can satisfy strict performance guarantees, thus this service category provides deterministic service. Non-real time applications can be supported by the UBR service category, here the sources do not declare any traffic parameters and this is effectively a best effort service. CBR and UBR are two very simple services which can accommodate the vast majority of applications, however by utilising the features of applications discussed in previous sections, service categories can be introduced which act to increase the utilisation and statistical multiplexing gain of the network.

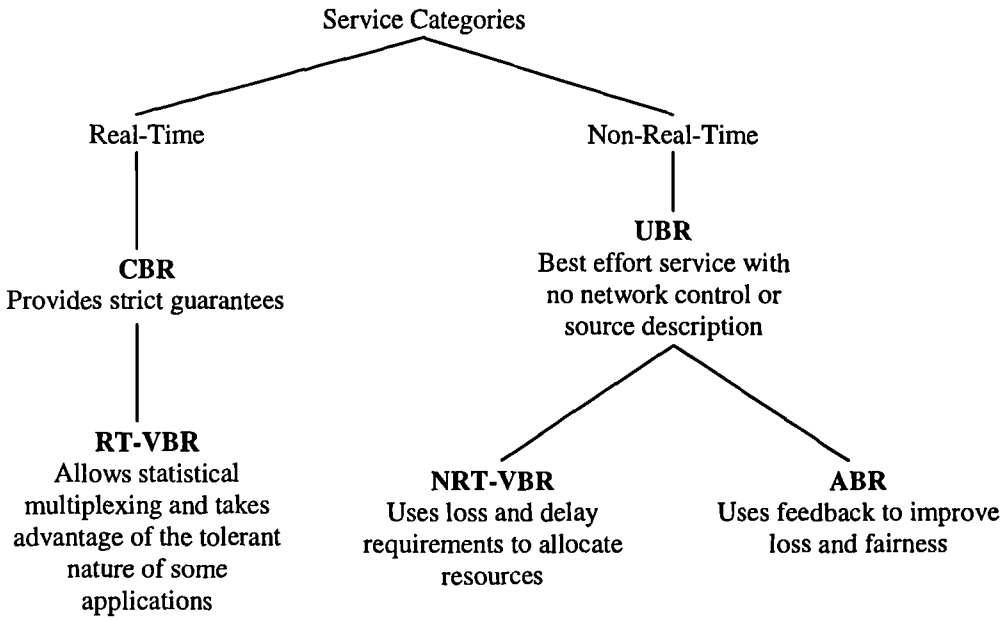

Figure 1.1:Architecture of ATM service categories

The CBR service category can be extended to take advantage of the bursty nature of traffic. This service category is called the real-time variable bit rate (RT-VBR) service category and provides a statistical service to tolerant applications by exploiting statistical multiplexing. The QoS for some non-real-time applications can be improved by differentiating these packets from the basic UBR packets. These non-real-time applications fall into the non-real-time variable bit rate

(NRT-VBR) service category, and requirements such as mean delay and loss rate are specified in order to allocate resources. A final service category called available bit rate (ABR) does not provide any guarantees and it attempts to minimise cell loss at the expense of delay by adopting a rate-based flow control strategy, it provides a service somewhere between UBR and NRT-VBR. The ATM service model architecture is shown in Figure 1.1 [3].

1.3.2.2 IntServ Service Model

As with ATM, IntServ will offer a service capable of providing mathematically provable deterministic guarantees to intolerant real-time applications, this will be called guaranteed service [8]. Another service category is the controlled-load service [9] which provides each flow with a QoS that closely approximates the QoS that same flow would receive on an under loaded network. This service category uses measurements to limit the load on the network in order to ensure that the desired service is received. Tolerant real-time applications are expected to fall into this service category and although no performance guarantees are made, either exact or approximate, the source must still declare its traffic characteristics together with some indication as to the desired QoS such that an admission decision can be made. The controlled-load service increases the network utilisation compared to guaranteed service by adopting greater statistical multiplexing at the risk of failing to meet delay requirements. The final service category is best-effort which is the same service currently offered by the Internet, however it has been suggested that there should be different classes of best effort with some classes receiving better service than others. The proposed architecture for IntServ is summarised in Figure 1.2

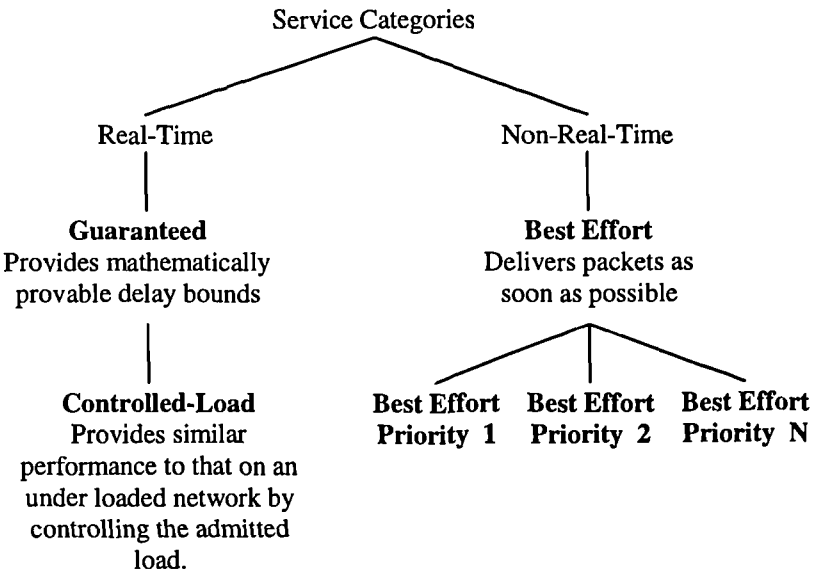

Figure 1.2:Architecture of IntServ service categories

Other service categories have been suggested and in time they might be integrated into the network architecture. In particular the predictive service in which the network supplies a fairly reliable, but not perfectly reliable, performance guarantee has received considerable support [4][5]. Using measurements of the current load, the network accepts connections it believes it can support and the occasional performance violation will be tolerated by the user in return for the presumed lower cost.

1.3.3 Congestion Control

As already mentioned, statistical multiplexing gives rise to the problem of congestion when demand exceeds network resources. Congestion control techniques can be classified as either reactive or preventative (sometimes called proactive). In a reactive approach, control is usually at the source and various feedback signals from the network or receiver are used by the source to detect congestion and react accordingly, usually by lowering the transmission rate. Such reactive control procedures operate on a time scale of the order of one round trip time (RTT), since the time between when congestion is detected and when the congestion signal is received is of the order of one RTT. An important parameter related to reactive control procedures is the delay-bandwidth product, which is the product of the RTT and the link bandwidth. This is the amount of data that can be transmitted by the source between when congestion is detected and a congestion signal is received, therefore this is the amount of data which may be lost due to congestion. Since the RTT depends upon distance and the speed of light, as link capacities increase so too will the delay-bandwidth product, thus leading to greater potential congestion. For this reason, more attention has been focused onto preventative control schemes. Three key areas of preventative control have been identified which must be incorporated into any possible solution to the congestion control problem, these are admission control, traffic enforcement and queue management [10], and crucial to each of these are the traffic characteristics declared by the source. Since it may still be possible for congestion to occur a fourth key area of reactive control is also identified.

The role of admission control is to determine whether there are sufficient resources such that the performance requirements of the connection seeking admission can be satisfied without degrading the performance of existing connections to unacceptable levels. The purpose of traffic enforcement or policing is to ensure that each connection is not violating its traffic characterisation declared at call set-up upon which the admission decision was based. Connections violating this may cause performance degradation for other connections together with their own, and the policing mechanism must enforce the original descriptor by carrying out

particular actions which may include dropping, delaying or marking packets. Without proper control, the interaction of several traffic streams may adversely affect network performance and it is the purpose of a queue management scheme to provide local short term congestion control. The role of a queue management scheme is essentially threefold, it must control which packets get transmitted, when packets get transmitted and which packets get discarded, these actions govern throughput, delay and loss, respectively. It is likely that preventative control schemes alone will not be sufficient to eliminate congestion in integrated services networks altogether, therefore when congestion does occur it is necessary to react accordingly. Generally, when congestion is detected, sources are requested to reduce their transmission rate until the congestion is cleared. As previously mentioned though, the high delay-bandwidth product may limit the effectiveness of such schemes, nevertheless, they are required as a safeguard mechanism.

Figure 1.3 summarises the time scales on which congestion control functions operate. Also indicated are other network control functions such as resource provisioning, which determine the physical quantities of equipment to be placed in the networks, and routing which determines the path packets take in order to reach the destination.

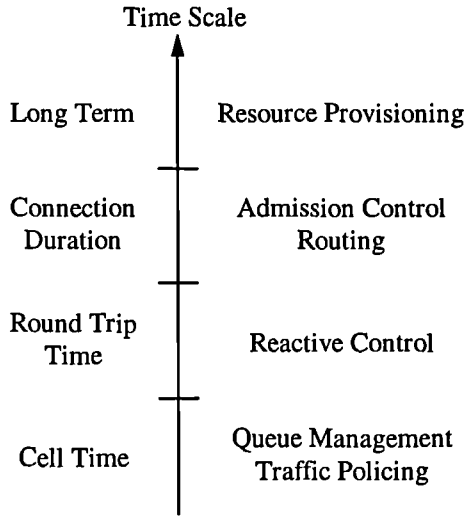

Figure 1.3:Time scale of various network control functions

1.4 Outline of the Thesis

The difficulties associated with accurate traffic characterisation, achieving high network utilisation and providing QoS guarantees, together with the sheer complexity and diversity of future integrated services networks makes the task of network control a difficult problem. It is the aim of this thesis to investigate the use of learning algorithms applied to the problem of

network control since they possess several features which may be beneficial, these include learning capabilities, high computation rates and a lack of a dependence on mathematical models.

In Chapter 2 we provide a state-of-the-art survey of traffic management and congestion control techniques in integrated services networks, highlighting the various problems associated with them. We continue by discussing various existing applications of learning algorithms to the problem of network control in Chapter 3, in particular we focus on the use of neural networks, fuzzy logic and stochastic learning automata. Chapter 4 investigates the suitability of neural networks to perform traffic prediction and contrary to popular belief we conclude that accurate prediction of real traffic on short time scales is impossible. We propose a new adaptive scheduling mechanism based on the use of stochastic learning automata in Chapter 5 and we show that it is able to perform well over a wide range of traffic conditions, whereas other static schemes only perform well in certain circumstances. In Chapter 6 we propose a novel admission control scheme for the adaptive scheduling mechanism, this is based on a neural network which is able to learn the required network resources despite there being no mathematical model available. Finally in Chapter 7, a summary of the thesis is given and areas for future work are identified.

Chapter 2

Traffic Management and Congestion Control in Integrated Services Networks

Traffic management and congestion control in communication networks deals with controlling network resources to prevent the network from becoming a bottleneck. In the previous chapter we highlighted that due to the high delay-bandwidth product of future high speed networks, preventative congestion control techniques are likely to dominate. Fundamental to preventative control is the way in which a traffic source specifies its characteristics and we begin this chapter by reviewing several methods of characterising and modelling network traffic. We continue by reviewing in turn the four major areas of congestion control, namely traffic enforcement, queue management, admission control and reactive control, and we conclude the chapter with a summary.

2.1 Traffic Characterisation and Source Modelling

Accurate traffic characterisation and modelling is important for several reasons. Firstly, in the context of network design and planning it provides an opportunity to study the behaviour of networks under a variety of conditions through analysis or simulation. This approach is less expensive and more flexible than experimental testbeds, thus speeding up the research process for novel network technologies. Secondly, in the context of congestion control, accurate traffic characterisation and modelling allows for efficient allocation of network resources such that QoS requirements can be satisfied.

Based on the declared traffic characteristics and requirements of a new call, an admission controller must decide whether or not to accept the call and, if accepted, must allocate appropriate network resources. The traffic characteristics of an application are the minimum set of parameters that a user can expect to declare while providing the network management with as much information as possible to effectively control network traffic and achieve high utilisation

[11]. However, efficient statistical multiplexing requires detailed knowledge of the traffic and this contradicts the requirement of a simple set of parameters, therefore the most suitable set of parameters remains an open issue [11]. Simple parameters include the peak rate and mean rate, more informative parameters include the mean burst length, index of dispersion, rate variance and autocorrelation, however these more sophisticated parameters may be difficult to estimate and enforce.

The traffic parameters are used to define a traffic model for the source, however due to the uncertainty surrounding the traffic parameters, together with a lack of understanding of the behaviour of some traffic sources there is no consensus as to the most appropriate traffic models. However, in [12], the authors identify several fundamental requirements of any traffic model. Firstly, the model parameters should be such that they can be inferred easily from the declared traffic characteristics. Next, the model should characterise the traffic as accurately as possible such that admission control algorithms do not over estimate the required resources. Finally, the model must be policeable so that the network can enforce a sources traffic characterisation.

In order to provide a deterministic service, deterministically bounded traffic models must be used to provide hard upper bounds on the quantity of traffic transmitted in a given time interval [13]. Deterministic or stochastic traffic models may be used to provide a statistical service, although stochastic models are more popular since their statistical properties allow for a greater statistical multiplexing gain. Traditional stochastic traffic models are based on stochastic processes such as Markov modulated processes [14][15], however recent traffic measurement studies have revealed that it may be more appropriate to model traffic using stochastic self-similar models [16][17].

2.1.1 Deterministic Traffic Models

Deterministic traffic models have recently been applied to the problem of providing deterministic delay guarantees to traffic with strict performance objectives. These bounds are determined using worst case arrival patterns and they provide upper bounds on the sources packet arrivals. If the actual traffic arrival process of a connection is given by A, such that $A[\tau, \tau+t]$ denotes the arrivals in the interval $[\tau, \tau+t]$, an upper bound on A can be given by the function A^* if for all $\tau \geq 0$ and $t \geq 0$ the following holds,

$$A[\tau, \tau + t] \leq A^*(t) \tag{2.1}$$

The function $A^*(t)$ is the traffic constraint function and is defined by the parameterised model declared by the source. In the (X_{min}, X_{ave}, I, s) model proposed in [6], X_{min} is the minimum packet inter-arrival time, X_{ave} is the maximum average packet inter-arrival time over any time interval I and s is the maximum packet size. The model proposed for ATM is the (PCR, SCR, MBS) model [18] in which the peak cell rate (PCR) is defined as the reciprocal of the minimum inter-arrival time, the sustainable cell rate (SCR) is the reciprocal of the average inter-arrival time and the maximum burst size (MBS) is the maximum number of cells that can arrive back to back at the PCR. The (σ,ρ) model [13] describes traffic in terms of a rate factor ρ and a burstiness factor σ such that the during any interval t, the traffic from the source is less than $\sigma+\rho t$.

The bounds associated with the models already mentioned are interval-length independent and a more accurate characterisation may be achieved by bounding the arrivals using intervals of different lengths. Such models are able to capture the intuitive property that as the interval length increases, the rate is bounded by a value nearer to the long term average rate. The (σ,ρ) model can be extended in this way to give the $(\vec{\sigma},\vec{\rho})$ model [19]. By maintaining a number of (σ,ρ) pairs, the amount of traffic in the time interval t is restricted to $\min_i\{\sigma_i+\rho_i t\}$, this leads to a traffic constraint function shown in bold in Figure 2.1.

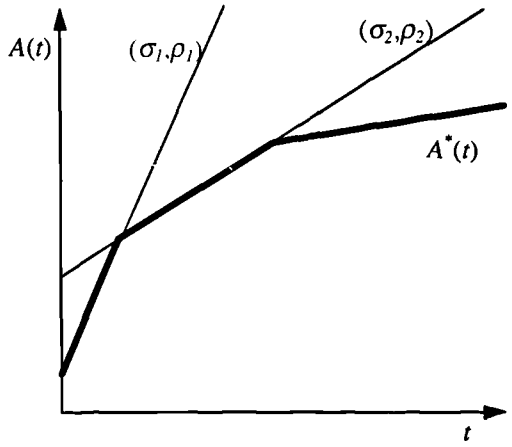

Figure 2.1:Traffic constraint function for the $(\vec{\sigma},\vec{\rho})$ traffic model

A similar approach is adopted in the D-BIND model [20] which characterises sources via multiple rate-interval pairs (R_k, I_k), where the rate R_k is a worst case arrival rate over the interval I_k. The authors show that such a characterisation allows for increased utilisation whilst still providing delay guarantees compared to interval independent traffic models such as the (X_{min}, X_{ave}, I, s) model in [6]

Deterministic traffic models have the advantage that they can be policed easily, thus ensuring that misbehaving sources do not adversely affect the performance of well behaved sources. Moreover, only deterministic models can be used to provide deterministic performance guarantees. However, an important drawback of deterministic traffic models when the goal is a statistical service is that they cannot characterise many of the statistical properties of the source, thus limiting the potential statistical multiplexing gain and consequently network utilisation.

2.1.2 Stochastic Traffic Models

In order to overcome the limitations of deterministic models, stochastic models can be used to increase network utilisation by exploiting statistical multiplexing. Although stochastic models cannot be used to provide deterministic performance guarantees, they can be used to provide statistical guarantees. Stochastic models fall into two categories, those based on stochastic processes and those based on stochastic bounds.

2.1.2.1 Stochastic Processes Models

2.1.2.1.1 Renewal Models

Renewal models have a long history because of their mathematical simplicity. In a renewal traffic process the inter-arrival times are independent and identically distributed, although this distribution may be general. A typical renewal process is a Poisson process in which the inter-arrival times follow an exponential distribution and such a model is completely characterised by a mean arrival rate. The discrete time counterpart of a Poisson process is a Bernoulli process in which the inter-arrival times are geometrically distributed. The single parameter describing these processes lead to analytical simplicity, however it is expected that bursty traffic will dominate integrated services networks [21] and renewal models are unable to capture the correlations present in such traffic, since by definition the inter-arrival times are independent.

2.1.2.1.2 Markov Models

Unlike renewal models, Markov models introduce dependencies into the arrival times and can consequently capture traffic burstiness. Markov models are based on a finite state machine and the process that governs the arrival process is completely determined by the current state. The sojourn time in each state follows an exponential distribution and the state changes are governed by a transition matrix $\mathbf{P}=p_{ij}$, where p_{ij} is the probability of a transition from state i to state j.

A typical Markov traffic model is a Markov Modulated Deterministic Process (MMDP) in which

packets are generated with constant inter-arrival time d_i, given that the current state is i. Variable bit rate voice is typically modelled as a two-state MMDP which alternates between exponentially distributed periods of activity and silence [14][15], this model is commonly referred to as the on-off traffic model. The mean rate, peak rate and mean burst length are sufficient to define this model and this simplicity allows it to be used to model general bursty traffic sources, moreover, its analytical tractability makes it the most popular model used in performance evaluation work [22]. Another popular Markov model is a Markov Modulated Poisson Process (MMPP) in which the arrivals follow a Poisson process with mean rate λ_i, given that the current state is i. A two-state MMPP has been used to model the aggregate arrival process of a number of voice sources [14] and the parameters of the MMPP are chosen such that the mean arrival rate, variance-to-mean ratio of the number of arrivals in $(0, t_1)$, long term variance-to-mean ratio of the number of arrivals and the third moment of the number of arrivals in $(0, t_2)$ are matched to the actual aggregate process.

More sophisticated Markov models have been used to model video sources. For example in [23] a Markov chain is used to model a low activity video source such as a person speaking into a camera. The state of the model determines the bit rate and the inactivity associated with these sequences leads to no abrupt changes in bit rate and for this reason state transitions are only allowed between adjacent states. The model parameters are chosen such that the mean, variance and autocovariance of the model match the actual video stream. A two-dimensional extension to this model is proposed in [24] to model high activity video sequences, with the extra dimension being responsible for large jumps in the bit rate which correspond to scene changes. An important consideration however with sophisticated Markov models is that they often involve many parameters and sometimes there is no apparent connection between them and simple statistics associated with the traffic source [25][26].

2.1.2.1.3 Self-Similar Models

Recent high resolution traffic measurement studies have revealed a fractal behaviour of traffic. These studies include the analysis of hundreds of millions of packets transmitted over an Ethernet LAN [27], the analysis of a 2-hour long VBR video trace [28] and the analysis of wide-area TCP traffic [29], in each case the traffic has been shown to display structurally similar traffic bursts across all time scales, from a few milliseconds to minutes and hours. This is called the self-similar phenomena and a critical characteristic of self-similar traffic is that there is no natural length of a traffic burst. Another important property of self-similar traffic is long range dependence which is apparent in the form of a hyperbolically decaying autocorrelation function,

this differs from the exponential decay of conventional traffic models. The degree of long-range dependence is measured using the Hurst parameter, H [27].

Methods of generating self-similarity include Fractional Gaussian Noise (FGN) and Fractional Autoregressive Integrated Moving Average (F-ARIMA) processes [30]. A self-similar model based on Fractional Brownian Motion (FBM) is proposed by Norros [31] to generate the total amount of traffic arriving at a system. The model is relatively simple in that only three parameters are required, namely the mean rate, the peakedness (variance to mean ratio over a unit interval) and the Hurst parameter. Moreover, the queue length distribution is derived and it is shown to be much heavier than the exponential decay predicted by traditional traffic models. Despite this, analytic results for self-similar traffic models are still scarce and simulation studies assume a special importance. Techniques for the fast generation of self-similar traffic include the Random Midpoint Displacement (RMD) algorithm [16] and Fast Fourier Transform (FFT) method [17].

In order to demonstrate the impact of long range dependence we have carried out an experiment to compare the behaviour of a queue, with a constant service time, using four different traffic streams. The first traffic stream is generated from a real trace obtained from an Ethernet LAN at Bellcore [32], further analysis of the trace can be found in [27]. In order to compare all streams we partition the trace into 10ms intervals and assume that the arriving data in these intervals is transmitted as ATM cells such that the cells are evenly spaced throughout the 10ms interval. The second trace is a randomly shuffled version of the original Ethernet trace, this shuffling has the effect of removing any long range dependence whilst maintaining exactly the same inter-arrival time and packet length distribution, the shuffled trace is then transformed into an ATM cell stream as before. The third trace is generated at random with the number of bytes in each 10ms period following a Poisson process with the same mean as the original Ethernet trace. The final trace is generated by the FFT method [17] such that the Hurst parameter and the mean arrival rate of the original Ethernet stream are matched. A plot of the utilisation of the queue against mean delay is given in Figure 2.2. Clearly the original Ethernet trace results in much greater delays than the shuffled trace and the trace generated using a Poisson process, this difference is due to the long range dependence in the original. The trace generated by the FFT model matches the original trace closely and it appears that this method of generation is reasonably accurate.

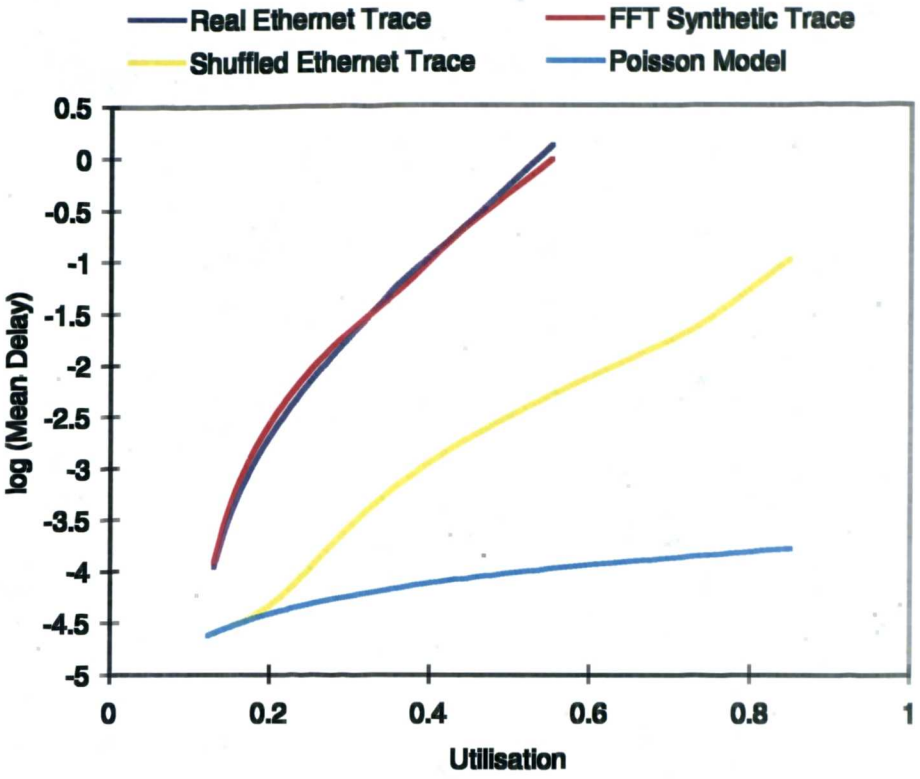

Figure 2.2:Comparison of performance of various traffic streams

2.1.2.1.4 Other Stochastic Models

Autoregressive models have been applied to model video sequences in [23] and [25]. The disadvantage with these models is that although they may be accurate compared to actual measurements they are unsuitable for analytical studies of queueing models, therefore they are used mostly in simulation studies. Transform-expand-sample (TES) models exhibit similar intractable properties and have been proposed to fit both the marginals and autocorrelations of empirical data [21].

In general, traffic models based on stochastic processes have the advantage that they may achieve higher network utilisation by exploiting the statistical properties of the sources, however they suffer from a number of disadvantages. For instance, the more straightforward traffic models are generally not powerful enough to capture the burstiness and important time correlations of realistic sources. By contrast, the more sophisticated models possess greater accuracy but often involve many parameters whose interpretation from physically meaningful quantities is unclear or they are too complex for tractable analysis. Also analytical difficulties often make the extension to heterogeneous sources and priority queueing systems difficult, both of which are required in integrated services networks. Furthermore, the diversity of traffic sources is likely to ensure that

a single stochastic traffic model is unlikely to represent all types of traffic and the use of several different models will complicate admission control and policing. Moreover, it is often impractical to determine whether a stochastic model is conforming to its specification. For example it is difficult to determine whether a Markov model is following a certain transition probability matrix, is close to its implied marginal distribution or has the required autocorrelation structure. Consequently, if the traffic does not conform then no guarantees can be made, furthermore, if admitted to the network a non-conforming stream may adversely affect the performance of other streams.

2.1.2.2 Stochastically Bounded Models

In order to overcome some of the difficulties associated with models based on stochastic processes, Kurose proposed a framework to characterise a source j by a family of random variables $\{B_j(t_1), B_j(t_2), \ldots\}$ [33]. In this model the random variable $B_j(t_i)$ is stochastically larger than the number of bits generated over any interval of length t_i by source j, where a random variable B is said to be stochastically larger than a random variable S if and only if $Prob(B>x) \geq Prob(S>x)$ for all x. A similar model to that of Kurose is proposed in [34], however in this model the bounding random variables are explicit functions of the interval length in order to better characterise the properties of the sources. An extension to the (σ,ρ) model in which the burstiness parameter, σ, is bounded by an exponential distribution is proposed in [35].

Although models based on stochastic bounds overcome many of the difficulties associated with models based on stochastic processes, problems still remain. Firstly, the problem of policing statistical characteristics still applies and secondly the determination of the various parameters, such as the rate variance, is not obvious. In order to overcome these limitations the parameters may be inferred or bounded based on policeable traffic characterisations such as the (σ,ρ) or the D-BIND model. In [36], the maximum rate variance is inferred from the worst case traffic arrival pattern of a source characterised by the D-BIND model. Based on this, statistical guarantees can be provided to a number of multiplexed sources whilst also achieving reasonable statistical multiplexing gain.

2.2 Congestion Control

As mentioned in the previous chapter there are three key areas of preventative congestion control, namely traffic enforcement, queue management and admission control. A further congestion control mechanism known as reactive control acts as a safeguard mechanism. Before

communication starts the client specifies its traffic characteristics and performance requirements. A set of admission control conditions are tested at each switch and the new connection is accepted only if its admission would not cause the performance guarantees of any connection to be violated. During data transfer each switch handles packets according to some queue management scheme such that local performance guarantees are satisfied. Therefore the admission control usually relies on the underlying queue management algorithms implemented in the network. Some researchers have proposed solutions for either admission control or queue management, however we, along with others [37][38], believe that a complete solution needs to specify both the queue management scheme and the associated admission control conditions. In order to satisfy any QoS requirements, it is crucial that the traffic parameters declared at call set-up are adhered to and it is the role of the traffic enforcement scheme to ensure this.

2.2.1 Traffic Enforcement

The purpose of traffic enforcement or policing is to ensure that each connection is not violating its traffic characterisation declared at call set-up upon which the admission decision was based. Connections violating this may cause a degradation in the performance of other connections together with their own, and the policing mechanism must enforce the original descriptor by carrying out particular actions which may include dropping, delaying or marking packets. As mentioned previously it is difficult to police statistical traffic descriptors, such as rate distribution or autocorrelation, and most of the policing mechanisms proposed in the literature control source parameters such as the peak and average bit rates and burstiness. Policing the peak rate is relatively straightforward, however policing the mean rate is a more difficult task since this is a statistical descriptor. In order to determine the true mean, the rate must be measured over the duration of the connection. However by the time it is determined that a connection does not conform, the connection will have terminated and the network already flooded with excess packets. Therefore measurements must be taken over shorter intervals which inevitably gives rise to a probability of incorrect policing decisions. Increasing the sampling period reduces this probability but also increases the reaction time of the mechanisms, therefore a trade-off exists.

The token bucket is an effective policing scheme and with appropriate selection of parameters it may enforce a range of traffic parameters, although a separate bucket is generally needed for each traffic parameter [11]. The basic idea is that for a packet to conform it must obtain a token from the token pool. If there is a token available, an arriving cell will consume one token and immediately depart. Tokens are generated at a constant rate, r, and placed in the token pool, however there is an upper limit on the number of tokens, b, allowed in the pool and tokens

arriving when this limit has been reached are discarded. The size of the token pool places an upper bound on the burst length and controls the number of packets that can be sent back to back. This upper bound is greater than the token bucket size since while packets arrive and consume tokens, new tokens are being generated. The scheme is summarised in Figure 2.3.

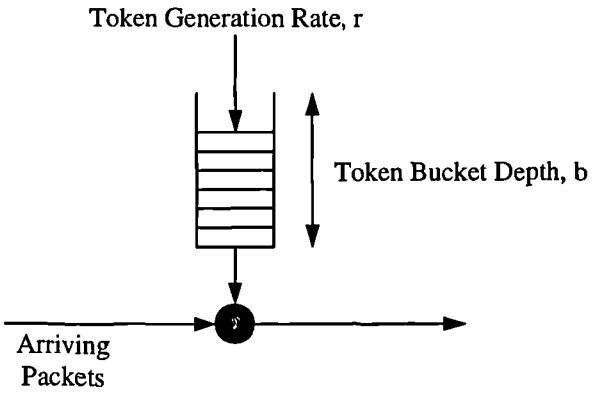

Figure 2.3:Token bucket traffic policer

The simplest source descriptor to police is one characterised by the (σ,ρ) model, since this simply requires the parameters r and b to equal σ and ρ respectively. The peak rate of a source may be policed by setting the token generation rate equal to the peak rate and the token bucket depth equal to the short term tolerable deviation from the peak rate. In the case of ATM the bucket depth is equal to the cell delay variation tolerance (CDVT). Similarly, the mean rate is policed by setting the token generation rate equal to the mean rate and setting the bucket depth such that short term variations from this mean are accommodated. The bucket depth represents the tolerable burstiness and in the case of ATM this is set equal to the burst tolerance (BT), where BT is defined as the maximum period a source can transmit at its peak rate.

Several actions are possible on the detection of a non-conforming packet in a token bucket policer. The most simple option is to simply discard the packet. Alternatively the packet could be buffered until a token is generated or it could be marked such that it receives low priority if congestion later arises. Buffering packets until tokens become available has the effect of shaping or smoothing the traffic stream and often this reduction in burstiness decreases the likelihood of congestion, however this buffering also has the effect of increasing delays.

Other methods of traffic enforcement include the jumping window, triggered jumping window, exponentially weighted moving average and moving window [11]. Such schemes suffer from the already mentioned difficulties associated with mean rate policing since the measurement interval

is determined by the size of the window.

2.2.2 Queue Management

Statistical multiplexing in packet networks causes packets from different connections to interact at each switch. Without proper control, these interactions may adversely affect network performance and it is the purpose of a queue management scheme to provide local short term congestion control. A queue management scheme essentially manages three resources : bandwidth (which packets get transmitted), promptness (when packets get transmitted) and buffer space (which packets get discarded) [39], which govern throughput, delay and loss rate respectively. Zhang [40] identifies four desirable characteristics any queue management scheme must possess, namely efficiency, protection, flexibility and simplicity. A scheme is more efficient than another one if it can meet the same performance guarantees under a heavier load. Furthermore, any discipline must also protect well-behaving connections from ill-behaving connections, network load fluctuations and unconstrained best-effort traffic. A flexible scheme is able to provide various delay, bandwidth and loss rates to different connections, and finally any discipline must be both conceptually simple to allow tractable analysis and mechanically simple to allow high speed implementation. These features describe the ideal queue management scheme, however, in reality we must trade-off certain features against others, for example a simple service discipline tends to be inflexible.

2.2.2.1 Packet Scheduling

The scheduling algorithm controls the order in which packets are serviced. The most simple scheduling algorithm is one that schedules packets in the order with which they arrive, generally known as First-In-First-Out (FIFO). This is the approach adopted in the current Internet and is very efficient when the traffic sources have roughly the same QoS requirements, since it treats all packets equally. However, for a heterogeneous mix of QoS requirements the inflexibility of FIFO makes it less efficient and results in low utilisation since admission control must be based on the requirement that each flow must satisfy the most stringent performance objective. Moreover, resources in a FIFO queue are shared in proportion to the offered traffic, thus a misbehaving source that injects excessive traffic into the network will result in the performance degradation of all the flows. Therefore FIFO is unable to perform protection, however protection may be achieved via external policing separate from the scheduler.

Adopting more sophisticated service disciplines may result in greater efficiency, flexibility and

protection, however one must remember that the service discipline must remain simple enough to implement in high speed networks. Next we shall describe a number of different approaches to scheduling.

2.2.2.1.1 Class Based Schedulers

A simple scheme capable of differentiating between traffic with different objectives is static priority (SP) scheduling in which packets of higher priority are always served before lower priority packets. For a heterogeneous mix of QoS requirements SP is more efficient than FIFO and this increase in utilisation comes from trading off the performance of the traffic streams with each other. For example if we consider a two-priority system which separates a real-time application from a delay tolerant non-real-time application, the high priority real-time application will experience a low delay at the expense of the non-real-time application, regardless of the non-real-time traffic load. This approach is thus able to protect the high priority stream from the low priority stream, however it is unable to perform the converse and will frequently lead to starvation of the non-real-time traffic. Furthermore, the inflexibility of SP leads to poor efficiency if the objectives are similar. For example, if the system consisted of two real-time applications with different delay requirements, the high priority stream may experience a much lower delay than is actually required and the low priority stream a much greater delay, therefore a mechanism beyond simple priority is required. An additional problem with SP is that if the QoS measure for the streams are different then it may be difficult to determine which should be given the higher priority.

With static priority the performance of the high priority traffic may be better than its QoS goals, therefore there is the possibility of improving the performance of the low priority traffic at the expense of the high priority whilst still meeting the objectives. An occupancy based scheme is proposed in [41] in which a threshold, T, is placed on one of the queues (Figure 2.4).

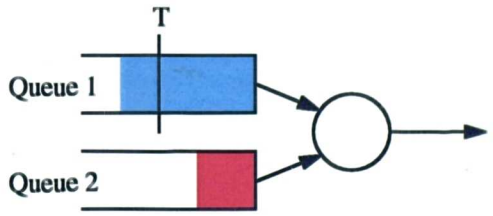

Figure 2.4:Occupancy based priority queueing system

If the occupancy of this queue is above the threshold then a packet from the head of that queue is served until the occupancy falls below the threshold, if the occupancy is less than the threshold

then the queues are served alternately. We have carried out experiments using this scheme and depending on the placement of the threshold it is possible to achieve varied performance. Figure 2.5 shows the mean delay of packets in queue 1 for various queue 1 and queue 2 traffic load ratios when there are 8 homogeneous Bernoulli traffic sources creating an overall load of 0.9. The performance falls between the two extremes of static priority, when the threshold is T=0 on either queue this is equivalent to static priority with the queue on which the threshold placed receiving high priority. If the correct threshold values are chosen then this scheme is simple and efficient, however the choice of threshold is not straightforward and becomes increasingly more difficult as the number of queues is increased.

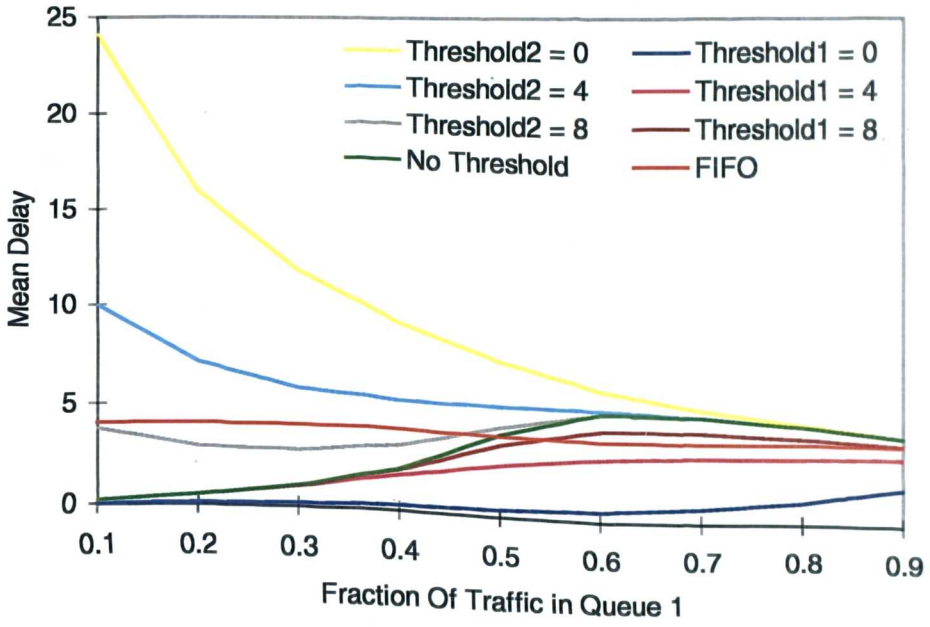

Figure 2.5:Mean delay of queue 1 traffic for various threshold values and traffic ratios

In [42] the authors propose an adaptive version of the threshold based scheme to support real-time video and voice. The thresholds are selected based on a sliding window measurement of the arrival rate from which the number of calls are inferred. The disadvantage with this scheme is that it assumes all traffic to follow a particular model with a given arrival rate and the thresholds are obtained from a look-up table, the contents of which were previously obtained from simulation. Another adaptive scheme is proposed in [43] in which associated with each queue is a weight and highest priority is given to the queue with the maximum weighted queue length. The advantage of this scheme is that it can quickly adapt to temporary overload, however the choice of weights remains a difficult problem.

2.2.2.1.2 Deadline Based Schedulers

The classic deadline scheduler is Earliest Deadline First (EDF) [44] in which arriving packets are assigned a deadline and transmitted in order of increasing deadlines. In [45] it is shown that EDF has optimal efficiency in that for a given set of connections, it can support delay guarantees that are at least as tight as those provided by any other scheduler. Since EDF selects packets according to their deadlines, its implementation requires a sorting mechanism. However, the complexity associated with maintaining a sorted queue may prohibit the use of EDF in high speed networks. A further drawback of EDF is that not all packets have deadlines and some performance objectives will be in terms of the fraction of late packets, therefore the loss of some packets will be more significant than others. This is not considered in EDF.

Several methods have been proposed to approximate EDF whilst requiring less complexity. For example in the Head-Of-Line with Priority Jumps (HOL-PJ) [46] scheduler, an arriving packet is placed in a FIFO according to its delay requirement and stamped with its arrival time. It is assumed that each packet in a particular queue has the same delay requirement and a packet may spend maximally T_j time slots in queue j before it jumps to the next highest priority queue j-1. The value of T_j is set to the difference between the delay requirement for the corresponding FIFO queue and that of the next higher priority FIFO. Therefore, under these assumptions, HOL-PJ is able to emulate the performance of EDF with much less complexity. However, the flexibility of providing a wide range of delay requirements is limited by the number of priority queues and increasing this flexibility will increase the associated overhead, furthermore, like EDF, the scheme is unable to deal with the relative importance of different packets.

A further problem associated with HOL-PJ is that packets need to be transferred between FIFO queues, this is avoided in the Rotating Priority Queues (RPQ) scheduler [47]. In a similar way to SP, RPQ is implemented with a set of FIFO queues, however this approach differs from SP in that the priority associated with each queue is modified (rotated) after fixed intervals. The shorter this interval the greater the flexibility in providing a wide range of delay requirements and the closer the scheme is able to approximate EDF, however this is at the expense of more FIFO queues.

A dynamic priority scheduler is presented in [48] which is similar to SP but can guarantee a minimum bandwidth to the lower priority streams. Associated with each queue is a counter which records the length of time a packet has been at the head of the queue. If this counter exceeds a particular threshold, the priority of the corresponding head of line packet is promoted

such that it will be selected for service in the next time slot. The counter is reset to zero after the head of line packet is served. As with the previous scheme the difficulty lies in selecting appropriate time thresholds for each queue.

2.2.2.1.3 Cost Based Scheduling

In order to take into consideration the relative importance of packets, several authors have advocated the use of cost functions. A cost function, $c_i(\tau)$, defines the cost incurred if packet i experiences a queueing delay of τ and the total cost incurred reflects the extent to which all performance objectives are met. For example, consider a performance objective defined in terms of a maximum fraction of late packets. If a delay of a packet exceeds its allowance A, a cost C is incurred, otherwise no cost is incurred. In this case the cost function is a step function and the less loss an application can tolerate the greater C should be. A heuristic called Maximum Utility Scheduling for Transmission (MUST) was proposed to minimise cost [49]. MUST orders all packets such that the cost would be minimised by transmitting packets in this order assuming no additional packet arrivals. Packets are transmitted in this order until another packet arrives, at this point the packets are reordered. Each time a packet arrives at a queue containing n-1 packets, an operation $O(n.n!)$ must be performed to determine the optimal order and the authors deem this complexity impractical.

In order to overcome this complexity Peha [50] proposes to evaluate a packets priority, which reflects the cost saved by transmitting the packet rather than keeping it in the queue, and transmit the packet with the highest priority. Since the evaluation of a packets priority is independent of the other n-1 packets then the complexity associated with this is approach is $O(n)$. However, the problem with this approach is the determination of the priority of each packet and the authors illustrate this with an example. Consider two streams from unrelated applications, stream 1 consists of packets for which the loss rate must not exceed 5% whereas stream 2 requires a maximum loss rate of 1%, but a stream 1 packet can tolerate more of a queueing delay before it is considered lost. In this situation the optimal ratio of costs is unclear and the authors suggest that the only way to overcome this problem is through experimentation or simulation.

In [51] it is identified that in addition to performing multi-criteria scheduling in which it is more important to achieve good performance for some packet streams than others, schedulers may be required to perform heterogeneous-criteria scheduling. In this scheme the performance measure associated with one stream may be different to that of another stream, for example one stream may have a maximum delay requirement whereas another may have a mean delay requirement.

Algorithms are proposed to perform this, however their complexity gives rise to impracticality and simply provide a bound on achievable performance.

2.2.2.1.4 Rate Based Scheduling

A class of service discipline called rate based disciplines have recently been proposed which provide a lower bound on the bandwidth allocated to each connection, therefore unlike the previous schemes these are naturally protective. These service disciplines can be classified into two categories depending on how they deal with sources sending packets at a higher rate than the bandwidth allocated.

- Rate allocating service disciplines will serve packets at higher rates providing that it will not affect the performance guarantees of the other flows.
- Rate controlled service disciplines will not serve packets at a higher rate under any circumstances.

Rate allocating service disciplines are work conserving, this means that the server never idles when there are packets waiting to be served. By contrast rate controlled service disciplines are non-work conserving, this means that the server may remain idle even when there are packets waiting to be served.

2.2.2.1.4.1 Rate Allocating Service Disciplines

The simplest rate allocating service discipline is round robin (RR). In this scheme separate queues are cyclically scanned, if an empty queue is selected for service then the next queue in the round is served. RR provides an equal share of the bandwidth to each queue, however by associating weights to each of the queues it is possible to provide different levels of service and further details of Weighted Round Robin (WRR) can be found in [52]. The disadvantage with WRR is that it requires integer ratios of the weights, therefore there is limited bandwidth allocation granularity. Moreover the choice of weights is a difficult problem and bandwidth allocation becomes unfair in the presence of non-uniform sized packets.

Another service discipline that provides rate guarantees is the Virtual Clock discipline [53] which aims to emulate a Time Division Multiplexing (TDM) system. Each packet is allocated a virtual transmission time, which is the time at which the packet would receive service if the server was actually performing TDM, and the packets are served in increasing order of virtual transmission time. This scheme has the disadvantage of being based on static TDM and the calculation of the

virtual transmission time is independent of the behaviour of the other connections. The delay of the packet is based solely on the entire history of that connection, therefore if a connection misbehaves it would be punished regardless of whether such misbehaviour affects the performance of other connections.

A number of schedulers which approximate fair queueing (FQ) or generalised processor sharing (GPS) [39] have been proposed and these attempt to provide each source with a share of the bandwidth as if it had its own dedicated server, whilst also ensuring that any unused bandwidth is shared in a fair manner. With GPS there is a FIFO queue for each connection and during any time interval when there are N non-empty queues, the server serves the N packets at the head of the queues simultaneously, each at a rate of $1/N^{th}$ of the outgoing link speed. GPS can be generalised in order to give different connections different service shares. In this case the system is characterised by N positive real numbers, $\phi_1, \phi_2, \ldots, \phi_N$, each corresponding to one queue. At any time τ, the service rate for a non-empty queue i is given by,

$$\frac{\phi_i}{\sum_{j \in B(\tau)} \phi_j} C \tag{2.2}$$

where $B(\tau)$ is the set of non-empty queues and C is the link speed. Therefore, GPS serves the non-empty queues in proportion to their service shares. GPS is impractical as it assumes that the server can serve all connections simultaneously and that the traffic is infinitely divisible. In reality, only one packet can receive service at a time and an entire packet must be served.

Packetised Generalised Processor Sharing (PGPS) or Weighted Fair Queueing (WFQ) [54] attempts to deal with these constraints by selecting for service the packet that would complete service first in a corresponding GPS system if no additional packets were to arrive. A more accurate emulation of GPS is achieved using Worst-case Fair Weighted Fair Queueing (WF^2Q) [55] which uses both the start times and finish times of packets in a GPS system. Rather than selecting for service the packet that will finish service the first as in WFQ, in WF^2Q the server only considers packets that have started (and possibly finished) receiving service in the corresponding GPS system. Both WFQ and WF^2Q guarantee that the accumulated service provided to each connection never falls behind the GPS system by more than one packet size, however only WF^2Q can ensure that the accumulated service is never more than one packet size ahead of GPS. Both WFQ and WF^2Q need to maintain a reference GPS server, which is not a trivial task. A less accurate but simpler approximation of GPS is Self-Clocked Fair Queueing

(SCFQ) [56] in which the finish time of the packet in service is used as the current virtual time value, however it is demonstrated in [40] that the inaccuracy can make SCFQ perform much worse than WFQ.

The Delay Earliest Due Date (Delay-EDD) [6] is an extension to EDF in that it provides protection. This is achieved by the server negotiating a contract with each source that states that providing the source obeys a given traffic specification, the server will provide a delay bound. The key here is that the server sets the deadline to be equal to the time at which the packet should have been sent had it been received according to the contract, this is simply the expected arrival time added to the delay bound. For example, if a client assures that it will send packets every 0.2 seconds and the delay bound at the server is 1 second, then the k^{th} packet from the client will get a deadline of $0.2k+1$.

Virtual Clock, WFQ, WF^2Q, SCFQ and Delay-EDD all use a similar sorted priority queue mechanism. In such a mechanism there is state variable associated with each connection and upon arrival of each packet the variable is updated and the packet is stamped with the value of this variable. Packets are served in the order of increasing priority index values, therefore a sorted priority queue must be maintained which adds to the complexity of the schemes. The computation of the priority index is straightforward in Virtual Clock and Delay-EDD, however for schemes that use the notion of virtual time in another reference queueing system, such as WFQ and WF^2Q, the computation is more complex.

For many of these service disciplines it is possible to calculate delay bounds providing that the traffic source conforms to a bounded model. It is also possible to provide delay jitter bounds, defined as the maximum delay difference between two consecutive packets, however these are loose in that they are simply equal to the delay bounds. The reason for this is that providing a work conserving discipline is adopted, it is possible for a packet to experience very little queueing delay in a lightly loaded network. By contrast, another packet can experience the maximum guaranteed queueing delay in a heavily loaded network, therefore the maximum difference in delay of two consecutive packets is equal to the maximum queueing delay.

2.2.2.1.4.2 Rate Controlled Service Disciplines

Since rate controlled disciplines can place an upper bound on the service rate of a flow, they are able to provide tighter delay jitter bounds than rate allocating disciplines. Some rate controlled disciplines use a regulator to control the traffic arriving at the scheduler (Figure 2.6) and these

regulators act to partially or completely reconstruct the traffic pattern at each switch so as to offset the effects of network load fluctuations and of the interactions between switches.

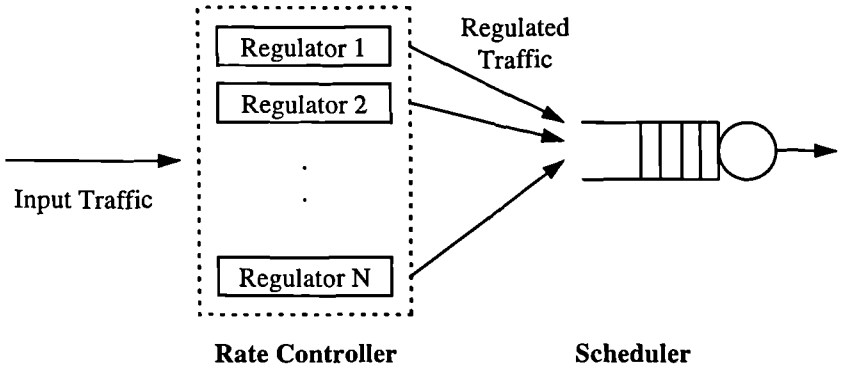

Figure 2.6:Rate-controlled service discipline with regulators

The Jitter Earliest Due Date [57] (Jitter-EDD) discipline extends Delay-EDD to provide tighter delay jitter bounds. After a packet has been served, a field in its header is stamped with the difference between its deadline and actual finishing time. The regulator at the entrance to the next switch then holds the packet for this period before making it eligible for service, this type of regulator is known as a delay-jitter controlling regulator. Although it completely reconstructs the original traffic pattern, adopting a delay-jitter regulator introduces the problem of additional information being required in the packet header. A rate-jitter controlling regulator holds the packet until the expected arrival time of the packet based on the traffic characterisation of the source, this notion of expected arrival time is similar in principle to that of Virtual Clock and Delay-EDD however the packet is not made available for service until this time lapses.

The Stop-and-Go service discipline [58] uses a simple framing strategy in which the time axis is divided into frames of constant length T. Bandwidth is allocated to each flow as a certain fraction of the frame and at each switch an arriving frame is mapped onto a departing frame by introducing a constant delay. This framing strategy introduces the problem of coupling between delay bound and bandwidth allocation granularity, since a larger frame size increases the delay bound yet is more flexible in allocating bandwidth. This problem can be overcome by introducing multiple frame sizes in a hierarchical structure. Hierarchical Round Robin [59] (HRR) is similar to Stop-and-Go in that it also uses a multilevel framing strategy, a slot in one level can either be allocated to a connection or to a lower level frame.

While Jitter-EDD can provide flexible delay bounds and bandwidth allocation, it is based on a

sorted priority mechanism which is difficult to implement. Stop-and-Go and HRR use a framing strategy to achieve simplicity, however this approach leads to a coupling between delay and bandwidth allocation. A Rate Controlled Static Priority (RCSP) server [60] consists of a static priority scheduler preceded by several rate controllers and attempts to achieve both flexibility in the allocation of delay and bandwidth, in addition to simplicity of implementation.

2.2.2.2 Packet Discarding

Queued packets occupy buffer space and some method of buffer allocation is necessary. One possible approach is to allocate a fixed amount of buffer space to each connection or class, however this allows for no sharing and is similar in principle to peak rate allocation of bandwidth. Usually the active connections share buffer space, however some packet discarding policy is required should the buffer approach capacity. Discarding policies fall into two different categories, namely partial buffer sharing and pushout [61]. It is generally accepted that the discarding policy is less important than the scheduling policy since buffer space is likely to be plentiful, for this reason we only briefly comment on the two approaches.

With partial buffer sharing an arriving low priority packet is only admitted if the state of a particular queue is below a given threshold. A global threshold may be placed on the overall occupancy or local thresholds may be placed on logical queues. Several priority levels may be supported by using nested thresholds where each priority has an associated threshold. The advantage of partial buffer sharing is its simplicity, however it is generally wasteful of buffer space since packets may be discarded even though buffer space is available, moreover the selection of appropriate thresholds is a difficult problem.

By contrast, pushout is space conserving and only discards packets when the buffers are full. If a high priority packet arrives at a full buffer it may overwrite (pushout) a lower priority packet. Although it may yield greater overall performance, pushout is much more difficult to implement than partial buffer sharing since it is necessary to keep track of every low priority packet in the buffer. This scheme is analogous to SP scheduling since it statically prioritises one class of packets over another. By adopting probabilistic pushout it is possible to tune the relative performance of the two classes by allowing a high priority packet to pushout a low priority packet with probability p. Experiments using probabilistic pushout have been carried out and Figure 2.7 shows the loss rate of both high (dotted) and low (solid) priority packets for various load ratios and probabilities. A probability of 1 is equivalent to pure pushout and a probability of 0 is equivalent to a scheme that simply discards the arriving packet, this is often referred to as

Last Come First Drop (LCFD). Clearly it is possible to achieve a range of performance with the various parameter values, however the selection of the optimal value depends upon the traffic environment and is a difficult problem.

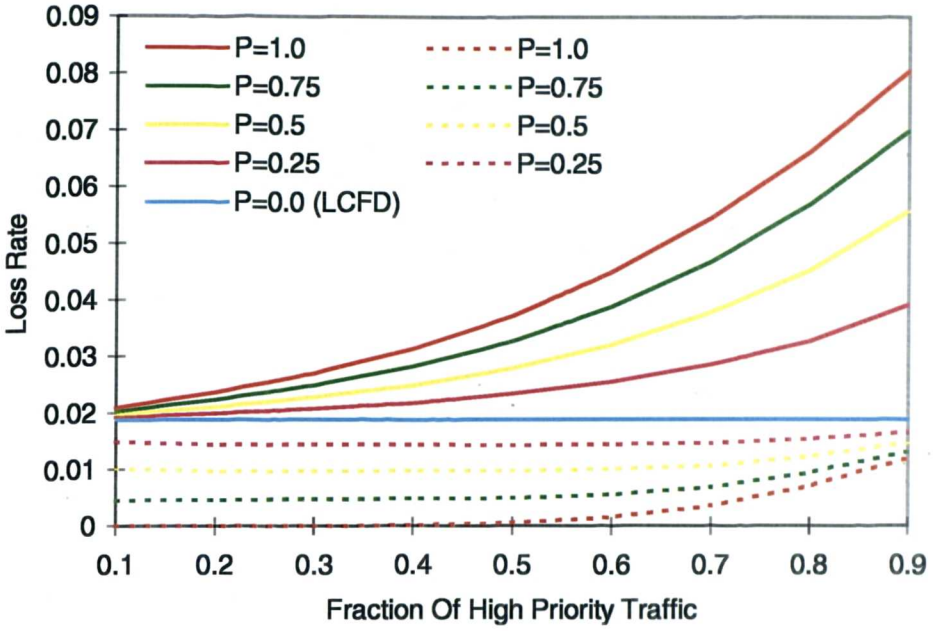

Figure 2.7:Loss rate of high (dotted) and low (solid) priority traffic for various load ratios and probabilities

2.2.3 Admission Control

Given that the source has specified its traffic parameters and assuming they will be adhered to, or at least effectively policed, the network must decide whether or not to accept the call. The role of admission control is to determine whether there are sufficient resources such that the performance requirements of the connection seeking admission can be satisfied without degrading the performance of existing connections to unacceptable levels. Any technique designed to carry out this function must do so in real-time and should attempt to maximise the utilisation of the network.

The QoS requirement of an application is generally specified on an end-to-end basis. Often the admission control policy assumes the method of nodal decomposition, this involves dividing the end-to-end requirement into a set of nodal requirements and checking whether each node has sufficient resources, however the most suitable technique for dividing the requirements is a difficult question. Nodal decomposition is not always necessary and networks employing rate based schedulers allow an end-to-end requirement to be mapped into a minimum bandwidth requirement. In this thesis we will generally assume nodal decomposition in which the QoS

requirements for each node are defined.

The simplest admission control policy is peak rate allocation and the connection is simply allocated its peak rate if it is available and rejected otherwise. This approach minimises the likelihood of congestion and is able to make deterministic performance guarantees, however it is very wasteful of bandwidth and does not take advantage of statistical multiplexing, especially when the sources have a high peak-to-mean rate ratio. A more aggressive policy, such as mean rate allocation, can accept more calls leading to greater network utilisation. However this introduces a greater risk of congestion and performance violations, especially in the presence of bursty traffic as there is a chance that a number of sources will burst simultaneously. Hence a trade-off exists between the achievable network utilisation and the confidence in providing performance guarantees. In general the amount of bandwidth allocated to a particular connection must be somewhere between its mean and peak rate.

Recently most of the literature has focused on admission control policies that provide deterministic guarantees under the assumption that the traffic sources are deterministically bounded. In order to provide a deterministic service it was assumed necessary to adopt the peak rate allocation policy [6] which as already mentioned results in severe under utilisation of the network. Recently however, it has been shown that this is not the case and delay bounds in a FIFO queue can be guaranteed even when the peak utilisation is greater than unity, providing that the incoming traffic is constrained by a bounded traffic model [62]. These firm guarantees are determined by assuming that all sources simultaneously generate traffic according to their worst case arrival process. Such a situation is unlikely to occur, however it is essential to make these assumptions in order to provide the required firm guarantees even though they generally result in under utilisation. Schedulability conditions have been mathematically proved for other service disciplines such as SP, EDF, WFQ and an overview of these can be found in [40][47].

In the context of providing a statistical service, most of the literature deals with traffic sources modelled as stochastic processes and only considers FIFO scheduling, since providing guarantees with more complicated scheduling mechanisms is difficult. A number of different approaches to providing guarantees in a FIFO multiplexer have been proposed which vary in terms of computational complexity, accuracy and traffic assumptions. Next we shall review a cross section of these.

A popular approach is the equivalent or effective bandwidth method [63]. Consider a single

source feeding a finite capacity queue, then the equivalent bandwidth of the source is defined to be the minimum amount of bandwidth required to achieve a desired QoS. As the QoS requirement becomes more stringent, the equivalent bandwidth increases from the source's average rate to its peak rate (Figure 2.8).

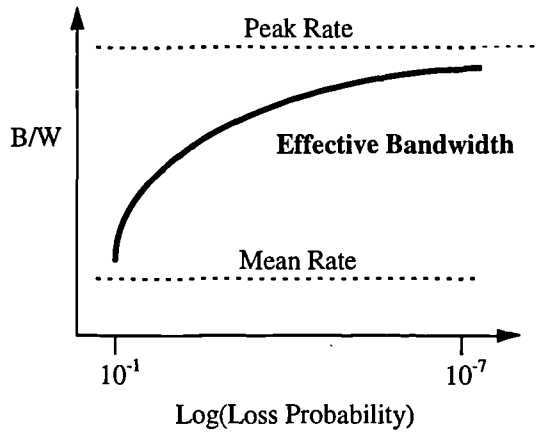

Figure 2.8:Effective bandwidth and QoS requirement

In [64], exact expressions are derived for the equivalent bandwidth of multiple 2-state MMDP fluid flow sources sharing a FIFO queue. The traffic is characterised by the triplet (R, ρ, b), representing the peak rate, mean rate and average duration of an active period respectively. Although exact, the expressions must be solved using iterative numerical techniques which make the method incompatible with real-time requirements.

Two approaches which approximate the exact method in [64] but require far less computational effort are described in [63]. The first approach simplifies the expression obtained in [64] in order to obtain a closed form solution for a single source. Given that the buffer is of size X, the bandwidth required such that a cell loss probability of ε is achieved is given by,

$$c = R\frac{y - X + \sqrt{(y-X)^2 + 4X\rho y}}{2y} \tag{2.3}$$

where $y = \alpha b(1-\rho)R$ and $\alpha = In(1/\varepsilon)$ [63]. It is assumed that the total bandwidth requirement of n multiplexed connections is equal to the sum of the effective bandwidths of the individual connections. This assumption may significantly overestimate the required bandwidth for the aggregate traffic since the effects of statistical multiplexing are not taken into account.

To capture the effects of statistical multiplexing, the second approach assumes that each

connection is characterised by its mean rate and variance. The problem is based on a bufferless fluid flow model and the goal is to find the required bandwidth such that the aggregate stationary bit rate exceeds this bandwidth with a probability of ε, the buffer overflow probability. It is assumed that the aggregate bit rate distribution is Gaussian with a mean and variance equal to the sum of the individual mean and variances respectively. The required bandwidth is given by,

$$c = m + \alpha\sigma \qquad \text{with } \alpha = \sqrt{-2In(\varepsilon) - In(2\pi)} \tag{2.4}$$

where m is the mean of the aggregate bandwidth, σ is the standard deviation of the aggregate bit rate and α is an approximation for the inverse Gaussian distribution [63]. This approach is very simple and requires little computational effort, however it does suffer from a number of limitations. Firstly the aggregate bit rate is exactly Gaussian only if the distributions of the individual connections are themselves Gaussian, and although this approximation is valid if there are a large number of connections each with similar parameters, in a realistic situation this is unlikely to be the case. Secondly, the possible gain from buffering is not taken into consideration since a bufferless model is assumed. In general, the first approach works well for a small number of sources whereas the second approach is superior for a large number of sources. Based on this the authors suggest using the lower of the two values to provide an upper bound on the cell loss. Equivalent bandwidth expressions for more general Markovian sources are derived in [65] and recently work has been carried out to determine equivalent bandwidth expressions for long range dependent traffic characterised by the mean rate, rate variance and Hurst parameter [66], however the estimation and policing of the Hurst parameter is likely to be extremely difficult.

In [67] a method is proposed to calculate the bandwidth required to satisfy a particular cell loss requirement. In the case of homogeneous sources the bandwidth is given by,

$$W = R(b, n, L)nB_m \tag{2.5}$$

where b is the burstiness defined as the peak-to-mean ratio, n is the number of sources, L is the mean burst length, B_m is the mean rate and $R(b,n,L)$ is a coefficient called the expansion factor. This scheme works on the assumption that in order to satisfy a particular loss requirement, the ratio of the effective bandwidth to the aggregate mean rate is the same providing that the burstiness and mean burst length are the same, regardless of the mean rate. This ratio is the expansion factor and to implement this approach the values of $R(b,n,L)$ must be precomputed for

a large number of (b,n,L) permutations. Another scheme based on a look up table is proposed in [68] and the advantage of such schemes is that they remove the real-time constraint by performing calculations off-line. Moreover, the calculations may be much more accurate and even take advantage of different traffic models. The disadvantage of these schemes however is that it is impossible to generate and store a table for every possible combination of traffic and QoS requirement.

The equivalent bandwidth approaches previously discussed assume knowledge of the mean burst length of the source, which may be difficult to declare and police. Several schemes have been proposed which simply assume knowledge of the peak and mean rate, an example of this is the virtual cell loss probability approach [69] in which the cell loss probability is calculated based on a bufferless fluid flow link. Cell losses only occur if the aggregate peak rate exceeds the link capacity and the virtual cell loss probability is the ratio of the excess traffic to the total offered traffic. The traffic is modelled such that source i is either generating at peak rate denoted by $(MAX)_i$ or they are idle. If the average bit rate of source i is denoted by $(AVG)_i$, then the probability that source i is active or idle is given by AVG_i/MAX_i or $1\text{-}AVG_i/MAX_i$, respectively. The probability density function of the traffic generated by source i is given by,

$$f_i(x) = \begin{cases} \dfrac{AVG_i}{MAX_i} & \text{if } x = \text{MAX} \\ 1 - \dfrac{AVG_i}{MAX_i} & \text{if } x = 0 \\ 0 & \text{otherwise} \end{cases} \tag{2.6}$$

Assuming that the sources are independent, the density function of the aggregate traffic, $q(x)$, is equal to the convolution of the individual density functions,

$$q(x) = (f_1 * f_2 * \cdots\cdots * f_N) \tag{2.7}$$

The convolution is then used to determine an upper bound on the loss probability and the observed loss probability converges to the virtual loss probability as the mean burst length of the traffic approaches infinity. However, the computational complexity involved with the convolution limits the real-time capabilities of such an approach, especially for a large number of sources. To overcome this a real-time algorithm is proposed in [70] which relies on a look-up table, however this approach does not suffer from the drawbacks of the previously mentioned

table approaches since it requires a finite memory, the contents of which are updated after each call acceptance or termination. The key advantage of the virtual cell loss probability approach is that it allows for easy policing, however the main drawback is that it overestimates the loss probability since it neglects the effects of buffering. A buffered extension is suggested in [71] which introduces a multiplication factor in the expression to calculate cell loss, however the authors conclude that there is no single factor ideal for every traffic type, and the determination of this factor is a difficult problem.

Providing statistical guarantees using scheduling disciplines other than FIFO is generally more complicated. If the assumed traffic model is a stochastic process then solutions for only the simplest scheduling disciplines such as SP have been derived, moreover these are based on quite simple stochastic traffic models. More success has been achieved when a stochastically bounded traffic model has been assumed, the most widely studied service discipline in this context is WFQ [72][73] with a exponentially bounded burstiness arrival process. Using a stochastically bounded traffic model, the authors of [34] derive stochastic delay bounds for FIFO and SP schedulers, however the complexity involved with calculating these delay bounds may be too great. In particular the delay bound calculation involves the summation of random variables which must be calculated via a convolution, this calculation may be too costly for on-line resource allocation, especially since these convolutions must be performed over multiple interval lengths. In order to overcome the difficulties involved with performing convolutions, the authors of [74] assume that the random variables are independent and therefore the Central Limit Theorem (CLT) can be applied. They assume that the sum of such random variables converges to a Gaussian distribution with a mean and variance equal to the sum of the constituent means and variances. This approach is only applicable when there are a large number of sources, furthermore it still requires a concise traffic characterisation in terms of the variance of the rate distribution. In order to overcome the problems associated with declaring the rate variance and policing stochastically bounded traffic, a method of inferring the rate variance based on the worst case arrival sequence of a D-BIND traffic characterisation is proposed in [36]. The assumed arrival sequence is one that maximises the rate variance and by applying the central limit theorem the tail of the delay probability is estimated. Such a scheme that allows significant statistical multiplexing gain whilst using easily policeable deterministic traffic models appears to be a very promising approach.

The idea of an admissible load region is proposed in [75] in which applications are divided into three classes and through off-line simulation the various combinations of calls that lead to QoS

guarantees being satisfied under the MARS scheduling algorithm are determined. This approach however has a fundamental drawback in that it assumes that each call in a particular class has the same QoS requirements and follows the same traffic model, which is unlikely to be the case.

The admission control policies already described are all parameter based in that the controller computes the amount of network resources required to support a given set of flows based on a priori flow characteristics. As already discussed, the efficiency of any scheme is reliant upon the information contained in the declared traffic parameters and it is often difficult to declare anything other than the simplest characteristics such the peak and mean rates, moreover it is also desirable to minimise the number of parameters to reduce complexity. By contrast, measurement based schemes rely on actual measurements of the current traffic to make an admission decision. Given the fact that traffic is generally dynamic in nature, guarantees made using measurement based policies can never be absolute and hence such schemes can only be used when the service commitment is a soft guarantee, such as in the predictive service. The scheme proposed in [76] only requires a source to declare its peak and average rate and information about the distribution of the traffic arrivals is continually updated through measurement. A more detailed discussion of measurement based admission control will given later in the thesis.

A different approach aimed in particular at VBR video is proposed in [77]. Here there is a continuous re-negotiation of traffic parameters with graceful degradation of QoS if the re-negotiation fails. When the rate of the compressed video changes significantly, where 'significantly' is defined by the individual application, a new traffic characterisation is passed to the network control. If the client is sending more traffic and sufficient resources are available to accommodate this then the re-negotiation is accepted, however if there are insufficient resources then the request is refused and the video source must reduce its compression factor such that the picture quality and bandwidth are reduced. The authors also propose an admission control scheme that bounds the re-negotiation failure based on the declared parameters of the new connections and the past behaviour of the existing connections. A similar approach is proposed in [78] where the bandwidth allocated to a particular traffic stream is dynamically adjusted based on estimates or predictions of the *interval effective bandwidth.* These approaches are based on two main arguments. Firstly, traffic is bursty over long time scales so therefore dynamically adjusting the bandwidth can achieve higher utilisation compared to a static worst case policy. Secondly, a static scheme requiring prior traffic characterisation is not suitable for applications which have no advanced knowledge of their parameters, for example live TV broadcasts. However, as with the measurement based admission control schemes already discussed these

schemes can only provide qualitative or soft QoS guarantees.

Many other admission control schemes have been proposed in the literature and a comprehensive review is beyond the scope of this thesis, the interested reader is referred to [79]. However, a number of key aspects must be considered before they can be applied to integrated services networks. Perhaps the most important point is the trade-off between the efficiency of the scheme and the accuracy of the traffic descriptors. In order to function efficiently a scheme must have a detailed knowledge of the traffic characteristics, however these characteristics are often very difficult to predict and police. Moreover, it is desirable to minimise the number of traffic parameters in order to reduce complexity, therefore given the current standpoint regarding traffic parameters it seems unlikely that high utilisation will be achieved using parameter based schemes. Another key point is that simple FIFO queueing is generally assumed and the extension to more sophisticated priority based queueing is often seen as further work [70]. The way in which these schemes would be adapted to cope with heterogeneous performance objectives would be to assume that several of these queues exist and are each served at a rate given by their effective bandwidth, however as we have already seen in the discussion of queue management schemes the accurate implementation of this in non-trivial.

2.2.4 Reactive Control

It is likely that preventative control schemes alone will not be sufficient to totally eliminate congestion in integrated services networks and when congestion does occur it will be necessary to react accordingly. Generally, when congestion is detected, sources are requested to reduce their transmission rate until the congestion is cleared. As previously mentioned, the high delay-bandwidth product may limit the effectiveness of such schemes, nevertheless, they are required as a safeguard mechanism. The reactive control scheme adopted by ATM standardisation bodies is Explicit Forward Congestion Notification (EFCN) [11]. In this scheme each switch in the network monitors its queue occupancies and when the queue size reaches a certain predefined threshold, all cells traversing the switch are marked until the occupancy falls below another threshold which indicates the end of the congestion period. A marked cell received at the destination indicates that there is a congested node along the path of this connection and the destination then signals back to the source requesting a reduction in transmission rate. EFCN is only effective when the congestion period is of the order of the propagation delay, since short term congestion could be over before any actions to alleviate it take place.

An alternative to EFCN is Explicit Backward Congestion Notification (EBCN) in which the node detecting the congestion informs the source directly, however this has not been accepted as an ATM standard since the overhead involved with generating a special cell may make the scheme impractical. A scheme proposed in [80] periodically sends time-stamped probe cells along the connection to measure the delay between source and destination. The main disadvantage with this approach is that these probe cells places an extra traffic burden on the network. Other reactive control schemes include adaptive rate control, in-call parameter re-negotiation and dynamic source coding [11].

2.3 Summary

In this chapter we have presented a review of various congestion control strategies likely to be adopted in integrated services networks and highlighted the various issues and problems which arise. The fundamental issue associated with integrated services networks is that of traffic characterisation. In order to achieve high utilisation it is necessary to have detailed information about the carried traffic and this inherently involves a large number of traffic parameters which may be difficult to estimate, enforce and analyse mathematically.

We began this chapter by discussing deterministic and stochastic traffic models. Although stochastic models allow for high statistical multiplexing gain, the parameters are difficult to estimate and police, and only the simplest queueing systems can be analysed mathematically. Deterministic model parameters are easier to estimate and police, and generally allow for more straightforward analysis, however deterministic models are unable to achieve the same levels of utilisation as stochastic models. We believe that the only practical method of traffic characterisation is through deterministic models and the loss of potential multiplexing gain is unavoidable.

Next we looked at various congestion control procedures and we found that generally the following three characteristics of any congestion scheme are directly correlated:

- Overall effectiveness

  In terms of the resulting network utilisation and the performance guarantees that can be made.

- Sophistication of the traffic model

  In terms of the number of traffic parameters and the accuracy of the model.

- Overall complexity

  In terms of real-time computation and implementation cost.

Therefore a reliance on traffic models that require a few simple parameters which are declared a priori and result in analytical tractability is likely to lead to severe under utilisation, since any control decisions must be based on the worst case arrival process that passes a policing test. To make matters worse the declared parameters may be conservative as the source may not know exactly how traffic will be generated, therefore the parameters are likely to represent upper bounds. If a refined traffic model is not used then either negotiated services or measurement based services must be used in order to increase utilisation. Unfortunately, these services can only provide soft performance guarantees and hence can only be applied to tolerant applications. Moreover, re-negotiated services require increased signalling overhead and measurement based services must make accurate predictions of future resource requirements using past measurements, which is a difficult problem.

The problems associated with network control have led researchers to consider the use of artificial intelligence (AI) techniques, since they possess a number of features which make them amenable to network control. For example, their prediction and learning capabilities may be useful in measurement based schemes and their ability to make fast intelligent decisions in situations where precise mathematical models are impractical or unavailable may overcome several of the real-time limitations associated with conventional methods. In the next chapter we provide a critical review of existing applications of AI to network control.

Chapter 3

A Critical Review of Artificial Intelligence for Network Control

In the previous chapter we demonstrated that high utilisation and guaranteed QoS are conflicting requirements in an integrated services network architecture due to problems associated with accurate traffic characterisation. Moreover, we identified that several of the proposed control strategies suffer from a number of drawbacks. Firstly, the complexity of the problems result in analytical intractability or huge computational demands which often ensure that they are infeasible in real-time. Secondly, in order to overcome the problems of real-time computation, approximations and assumptions are made which are unrealistic and hence limit accuracy. Moreover, some techniques are only suitable in a certain network domain and could not be applied to a more general scenario. Finally, several strategies rely on data that provides little useful information, hence static schemes based on this data are often very conservative. In order to overcome such problems the use of artificial intelligence (AI) techniques has received considerable support and in this chapter we present a critical review of applications of AI to network control.

3.1 Artificial Intelligence

Artificial intelligence is a relatively young discipline and is concerned with the automation of intelligent behaviour. This description is rather vague however, since intelligence itself is not very well defined or understood. Therefore the problem of defining AI becomes one of defining intelligence. There are many definitions of AI, however they can be broadly categorised into the four areas summarised in Figure 3.1 [81].

Systems that think like humans	Systems that think rationally
Systems that act like humans	Systems that act rationally

Figure 3.1:Definitions of AI

The definitions on the top are concerned with thought processes and reasoning, whereas the ones on the bottom address behaviour. The definitions on the left measure success in terms of human performance, whereas the ones on the right measure success against an ideal concept of intelligence i.e. doing the right thing.

Applications of AI include game playing, automated reasoning and natural language understanding. Although the range of applications is diverse, there is often a common reliance on techniques pertaining to knowledge representation and search. The function of any knowledge representation scheme is to capture the essential features of a problem domain and make that information accessible to a problem solving procedure. Search can be viewed as the systematic exploration of a state space which represent solutions to the problem.

Traditionally a purely symbolic paradigm has been used in AI to represent knowledge. Symbols refer to objects and relations, and the search mechanisms used to examine the space of the representation are often heuristic in nature. Although based on sound logical principles, a symbolic approach may be lacking. For example, an expert system may perform well in a certain domain, but should it encounter a problem outside of that domain it will typically be unable to suggest a solution. On the other hand, human experts will attempt to answer the problem to the best of their ability. In the context of network control, such deficiencies were made apparent in the previous chapter where admission control decisions were based on pre-computed calculations of various combinations of traffic, however the potential diversity of integrated services networks makes a comprehensive representation infeasibly vast.

A non-symbolic or computational intelligence approach to knowledge representation is based on the collective behaviour produced by the interactions of a number of simple components and views knowledge as being represented in patterns of interaction between these components. In this chapter we shall focus of three different areas of non-symbolic AI, namely neural networks, fuzzy logic and stochastic learning automata. Neural networks (NNs) are perhaps the best known application of a non-symbolic approach and consist of a large collection of single units which are capable of very little in isolation. A neural network is not programmed with information, but is instead trained by exposing it to large amounts of data and the connections between the units are updated in order to represent the knowledge. A non-symbolic approach is often adopted when humans do not have the knowledge and hence cannot encode it or the level of knowledge is below that of conscious knowledge, for example pattern recognition. The possible disadvantages with a non-symbolic approach include the lack of a firm logical base and the

general uncertainty surrounding them.

Although there are some problems associated with AI techniques, there are a number of important features common to many divisions of the field which make them attractive for tasks such as network control, these include [82] :

- The use of computers to learn or perform some other form of inference.
- A focus on problems that do not respond to algorithmic solutions.
- A concern with problem solving using inexact, missing or poorly defined information.
- Answers that are neither exact or optimal, but are 'sufficient' in situations in which traditional approaches are too expensive or not possible.

3.2 Neural Control of Networks

A neural network is an implementation of an algorithm inspired by research into the brain. The basic building block of a neural network is a single processing element, known as a *neuron*. The neurons are interconnected and the strength of these connections, known as *synaptic weights*, store the knowledge and determine how the neural network functions. The synaptic weights are either pre-programmed into the network or they are inferred by exposing the network to information and allowing the weights to be adjusted based on performance. A brief overview of the theory of neural networks can be found in Appendix A and a more comprehensive review of all aspects of neural networks can be found in Haykin [83].

The advantages often associated with neural networks include [84] :

- Adaptivity

  They take data and learn from it. Thus they infer solutions from the data presented to them, often capturing quite subtle relationships. Moreover, this approach may allow the solution of problems which are intractable using conventional methods.

- Generalisation

  They can correctly process data which is quite different from the data it was trained on. Similarly they can handle imperfect or incomplete data, thus providing a degree of fault tolerance.

- Non-linearity

  They can capture complex non-linear interactions between input variables, thus allowing application to many real world systems.

- Parallelism

  The numerous identical, independent operations associated with neural networks allow for fast simultaneous execution in hardware.

Despite these advantages they suffer from a number of drawbacks [83][84]. For example, it can be difficult to account for the results since unlike human experts they cannot answer the question 'why?'. Secondly, training methods are poorly understood and network topology design is very much a black art, relying more on trail and error rather than sound fundamental principles. Moreover, neural networks can consume huge amounts of computer processing time during training due to the amount of the training data and network size. Finally, they may suffer from the problems of over fitting and local minima. Over fitting occurs when a neural network learns too many specific input-output relations (i.e. it is overtrained) and is unable to generalise between similar patterns. The problem of local minima results from the fact that many neural networks are basically hill climbing techniques, hence there is a risk that every small change in the configuration of the network would result in an increased cost function, whereas a more radical change in the configuration would yield a smaller cost function.

The adaptability of NNs led Hiramatsu [85] to suggest a NN network control hierarchy in which the different levels of the hierarchy co-operate to achieve an effective network-wide control system. The different levels of the hierarchy correspond to the cell level, call level and network level aspects of control which work on the sub-millisecond, sub-second and weekly time scales respectively. The proposed method is summarised in Figure 3.2.

In this section we review various aspects of network control to which neural networks have been applied. These include admission control, link allocation, traffic policing, flow control and switch control.

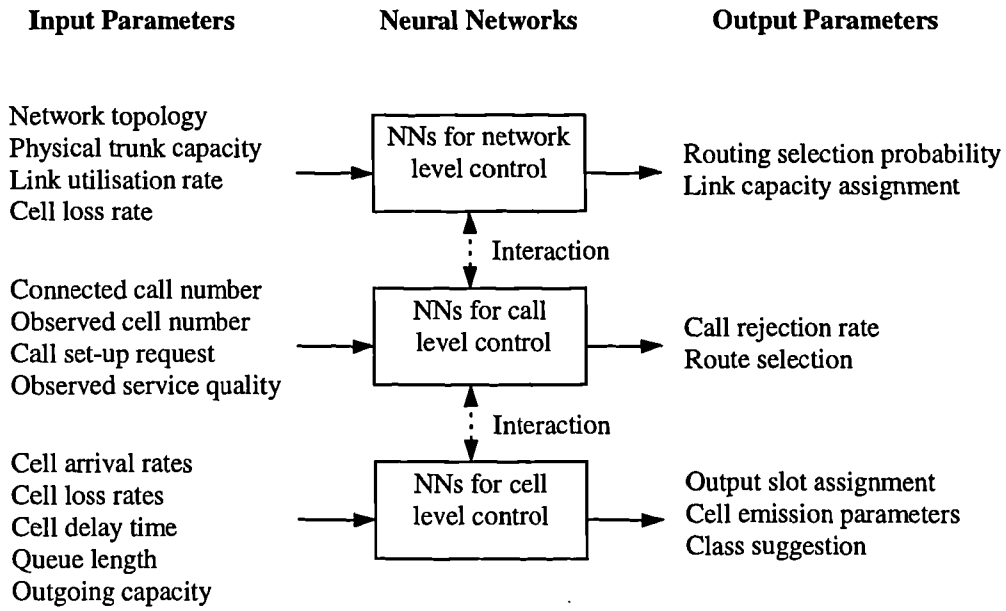

Figure 3.2:Input and output signals of neural networks in hierarchical ATM control [85]

3.2.1 Neural Admission Control

Admission control is the most popular network control problem to be addressed through the use of NNs. The general principle behind most of the schemes is that the neural network learns the behaviour of a multiplexer and bases the admission decision upon this. The input signals to the NN may consist of the observed multiplexer status, such as the packet arrival rate and the number of connected calls, and the traffic parameters declared at call set-up. Output signals may include the expected QoS and decision values for acceptance or rejection.

The proposed schemes differ in the type of NN used, the various inputs and outputs and how the network is trained. There are two approaches to training : off-line training and on-line training [86]. In the off-line approach the network is trained with data already obtained from analysis, measurement or simulation and this approach is suitable when the real system is exactly the same as the model. Usually however there are differences between the model and the actual situation. To account for these differences on-line training uses data obtained from monitoring the system in real-time and can therefore accommodate changes in traffic characteristics, however the continuous updating of the weights adds considerable overhead.

One approach to off-line training was proposed in [87] in which a NN is trained to learn the effective bandwidth of a number of multiplexed on-off sources given the desired packet loss rate. After off-line training, the NN was shown to be almost as accurate as the analytic solution given

in [64] yet it required far less computational effort and could hence be used in real-time. Moreover, it was shown to significantly improve upon Guerin's analytic approximation [63] also aimed at reducing the complexity.

Although off-line training may be useful in order to allow the weights to adjust to almost optimal values, it is often the case that continuous on-line training is required to improve accuracy and adjust to changing conditions. In [88] a learning control method is proposed in which a 3-layer fully connected NN trained using the back-propagation algorithm creates an admission decision function by learning the behaviour of a multiplexer. The shape of this function is expected to be correctly modified when the characteristics of the multiplexed calls change. Two approaches are proposed, in the single bit rate case the inputs to the NN are the recent packet arrivals and in the multiple bit rate case the inputs are the number of active connections. The output is in the range [0,1] and is denoted by $J(t)$, the call rejection rate $r(t)$ is determined by the following relationship,

$$\begin{aligned} r(t) &= 0 \qquad \text{if } J(t) < J_0 \\ r(t) &= 1 \qquad \text{if } J(t) \geq J_0 \end{aligned} \tag{3.1}$$

where J_0 is the call rejection rate threshold. Therefore if $r(t)$=0, all call set-up requests are accepted; otherwise all requests are rejected.

In the application to the single bit-rate case, in which each requesting call has the same traffic characteristics and requirements, the inputs to the NN are simply the previously observed values of the packet arrival rate. A number of on-line training methods are proposed, the simplest of which is to use the data observed during previous consecutive monitoring intervals. This approach has the disadvantage that the training pattern is likely to cover a small domain since the behaviour of the multiplexer will not change much in a short time. In the pattern table method, a large amount of observed data is stored in two tables, one for high loss rate events and the other for low loss rate events. Whenever there is a new observation, this replaces the oldest observation in the corresponding table and training takes place on data in a randomly selected table. Hence the possible training examples cover a wide domain including both high and low loss rate scenarios. An extension to this approach is the leaky pattern table method in which there is a mechanism for discarding patterns from the pattern table such that the NN does not continue to learn patterns which are old and no longer accurate.

In the multiple bit rate case, the connections are categorised into groups according to their traffic

characteristics, a simple way to define these groups is by application, such as voice, video and data. The status of the multiplexer is represented by (n_{vo}, n_{vi}, n_{da}), where n_{vo}, n_{vi} and n_{da} are the number of active connections using voice, video and data services respectively. The neural network learns the decision function, $J(t)$, based on the combination of n_{vo}, n_{vi}, n_{da}.

$$J(t) = F(n_{vo}, n_{vi}, n_{da}) \tag{3.2}$$

The authors of [88] conclude that the greater the diversity of the arrival process, in terms of bit rate fluctuations and number of bit rate classes, the more neurons are required in the hidden layer, since the decision function to learn is more complicated.

In [89], the authors argue that a better approach involves learning the relationship between the multiplexer status and a continuous variable, namely the expected performance, rather than a discrete indicator variable that determines whether performance will be above or below the required level, as in [88]. Because of this continuity, the authors feel that the NN is better able to interpolate and extrapolate to regions of the domain space not frequently experienced, therefore schemes such as the leaky pattern table are not required. This generality also provides flexibility in that a change in the required performance level while the network is operational does not require retraining. The scheme proposed in [89] does not adopt the back-propagation training algorithm since the performance measure is delay and adopting the common pattern mode training approach, in which weight updates occur after each observation, may not lead to convergence due to the volatility of individual packet delays. Instead, a rather large window of previous observations is used and all of these observations are considered in each weight update using a method of summarisation to minimise some objective function. This approach however appears to be very similar to batch mode training with the back-propagation algorithm in which the weight updates are based on the cumulative error function obtained from a number of observations.

If the performance measure is loss then an accurate mapping between the multiplexer status and small packet loss rates may be difficult to obtain unless the measurement interval is very long. The virtual output buffer method overcomes this problem by monitoring the performance of a virtual buffer [90]. A virtual buffer is a simulated buffer which has the same size and arrivals as the actual buffer, but has a lower service rate which obviously leads to greater loss. By extrapolating the loss rate from virtual buffers with smaller capacity, it is possible to estimate the loss in the actual buffer.

A hybrid approach to admission control is proposed in [91] in which a NN is combined with a tractable analytic approach. The admission decision requires a number of steps. Firstly an analytically simple calculation is performed to obtain an upper bound approximation of the expected loss rate. If this is below the target loss probability then the call is accepted. If the value is above the target then a NN, trained in that region, is used to decide whether or not to accept the call. The role of the NN is therefore to refine the upper bound estimate by applying knowledge learned from a much more complicated analytical model. The reason for this approach is that only training the NN in a subset of the state space speeds up the off-line training and allows more accurate and thorough training.

Instead of basing admission decisions on actual observed arrival rates as in [88], the authors of [92] propose to base decisions on the power spectrum of the arrival process since it contains information as to the correlation behaviour of the arrivals. The inputs to the three-layer NN are the necessary power spectrum parameters, such as the total average power, and the output is a decision as to whether to accept or reject a new call. The main problem with the NN admission control schemes proposed in this section is that they make questionable assumptions about the traffic, for example it is often assumed that the traffic characteristics of the calls are the same and in order to accommodate heterogeneous sources it may be necessary to use several NNs.

3.2.2 Neural Link Allocation

The link allocation problem arises during routing when a path consists of several parallel physical links. The objective of the link allocation function is to maximise the long term revenue whilst maintaining acceptable QoS. For example, consider the situation in which there is just enough available bandwidth on a link to support a high-bit rate connection. Providing that high-bit rate connections request admission with sufficient regularity, the controller should be capable of rejecting low-bit rate requests in order to retain sufficient bandwidth to accept a high bit-rate connection and hence maximise utilisation and revenue.

In the method called Back Propagation with Hypothetical Targets (BP_{HT}) [93], a delayed reinforcement learning approach is adopted in which a neural network is trained using a simple bipolar reward indicating whether the allocations were successful or not. The network maps a state vector to an M-dimensional output vector, where M is the number of possible actions. Next an action selector transforms this output vector into a binary action vector of the same dimension, where all values are zero except the one corresponding to the chosen link. The difficulty arises in training this NN since link allocations not only affect the current state of the network, but also

future states, which will also depend on future link allocations. So therefore it is difficult to determine at the time of allocation whether or not the chosen links were suitable. To overcome this it is assumed that the current allocation will eventually turn out to be a good one, therefore the action vector is a good hypothetical target and training using this would favour the action just taken and inhibit all others. This is called an optimistic target vector. Similarly a pessimistic target vector may be constructed under the hypothesis that the current allocation will turn out to be bad. The NN accumulates two possible weight changes for each of its weights until a feedback signal arrives which determines which set of weight changes to adopt. It is demonstrated that the NN is able to reach a level of performance comparable to conventional dynamic programming methods, however it is claimed that the approach is superior in that it does not make any assumptions about call inter-arrival and holding times.

An alternative approach is proposed in [94] in which a temporal-difference learning scheme is applied. It decomposes the link allocation task into a set of link admission control tasks which are modelled as semi-Markov decision problems and the neural network must estimate the expected reward given an admission request, where the reward is aggregate quantity of data transmitted, therefore the goal is maximise utilisation.

A slight variation on the link allocation problem is tackled in [95]. Here a method of integrating call admission control and link capacity assignment is proposed and the relation between the two mechanisms is shown in Figure 3.3.

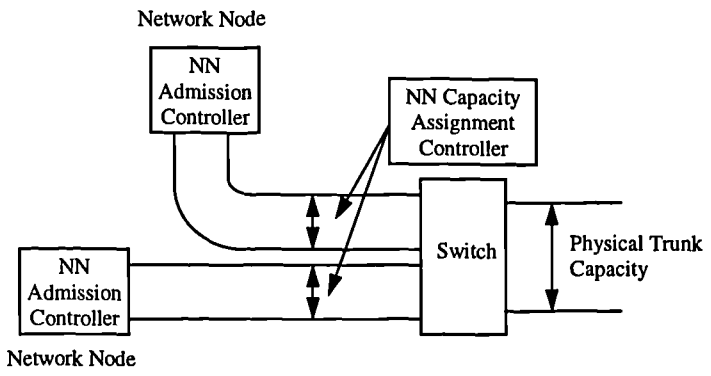

Figure 3.3:Call admission control and link capacity assignment

The scheme consists of three NNs, two perform admission control for the output link of each node and the other is used for optimising the link capacity assignment. The two output links share one physical trunk and the NN must allocate capacity to each link based upon the objectives, these include minimising the maximum call rejection rate, minimising the average call

rejection rate and maximising link utilisation. Therefore the purpose of the link capacity assignment NN is to optimise the link allocation based upon various requirements using estimates of the call rejection rate.

3.2.3 Neural Traffic Policing

Most conventional policing mechanisms attempt to police the peak and the mean bit rates of traffic. However, such mechanisms only enforce one parameter of the probability density function (PDF) of the traffic source. Moreover, there is a trade-off involved with policing the mean rate in that a long measurement interval is required in order to confidently judge whether the source is complying, however a long measurement interval leads to a slow response time to any violations. Policing mechanisms that attempt to police the PDF face the difficulty of complicated calculations of higher order moments. A policing mechanism using neural networks, called Neural Networks Traffic Enforcement Mechanism (NNTEM), was introduced in [96]. The authors claim that the NNs are able to learn the PDF of a traffic count process, one is trained off-line to learn the PDF of an ideal non-violating traffic source and the other is trained to learn violating traffic patterns from on-line measurements. The claim that the NN learns the PDF is in fact a false one and actually both NNs are simply predicting future values of the traffic count process from past history. The inputs to the NN are samples of the count process over some monitoring period and the output is a prediction of the count process in the next interval. By comparing the outputs of the two NNs an error signal is fed into a third NN which interprets it and outputs a signal to indicate how many packets to drop in the next interval. The weights of this third NN are tuned using reinforcement learning to minimise the error signal produced by the other two NNs and hence minimise the difference between the actual and an ideal traffic source.

A similar policing mechanism is proposed in [97], once again past arrivals are used as inputs to the NN and the output is a prediction of the traffic in the next interval. The number of cells to be discarded or marked is given by the difference between the number of measured and predicted cells. Therefore the neural traffic policers discussed are reliant on NNs performing traffic predictions, this application of NNs will be discussed in more detail later in the thesis.

3.2.4 Neural Flow Control

A neural network controller for flow control of video traffic was presented in [98]. The controller uses the buffer occupancy as a measure of potential congestion and if congestion is detected a signal is fed back to the source requesting a reduction in its traffic generation rate. The video

source reduces its transmission rate by increasing its level of coding, this results in lower picture quality. The inputs to the neural network are the buffer occupancy levels at previous observation periods and the output is a signal to indicate by how much the source should increase or decrease its coding rate. The back-propagation algorithm is used to train the NN and minimise some performance index which attempts to minimise the loss probability and maximise the level of coding. A similar scheme is proposed in [99] where this time the inputs to the NN are the previous traffic arrivals and the output is a prediction of the future traffic. The traffic prediction is then used to determine the future buffer occupancy which consequently determines the feedback signal, the overall scheme is summarise in Figure 3.4.

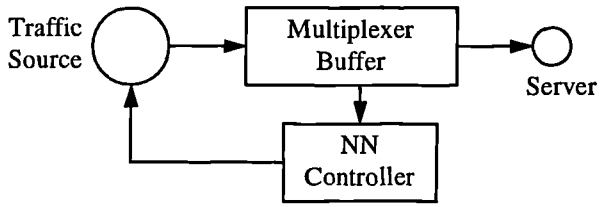

Figure 3.4:Neural network feedback controller

3.2.5 Neural Switch Control

In the applications of NNs already discussed, their learning and function approximation qualities have generally been utilised. By contrast, in their application to switch control they are generally used to perform optimisation and their most common task is the scheduling of packets in input buffered switches. Input buffered switches suffer from the problem of Head-Of-Line (HOL) blocking in which a packet in a FIFO queue has no chance to access an available output port because the packet ahead of it in the buffer is blocked from accessing its output port. The HOL blocking problem severely limits the achievable throughput in such switches [100], however in order to overcome this problem bypass queueing may be employed which allows packets to be transmitted when the leading packets are blocked. The problem of which packets to transmit is a maximal matching problem in which it is required to transmit the maximum number of packets, whilst also satisfying the constraints that only one packet may be transmitted from any input port and only one packet may arrive at any output port.

Using conventional algorithms, the solutions to such combinatorial optimisation problems require significant computation and are unlikely to meet the real-time requirements of high speed switches [89]. The parallelism associated with neural networks has prompted researchers to apply them to this problem. In [101], a Hopfield network is configured such that it acts to

minimise an energy function which encodes the goals and constraints and it is demonstrated that the NN can achieve near optimal performance at very high speeds, with sub-optimal performance occurring when the NN converges to local minima. The problem of priority is tackled in [102], in this approach instead of assigning a binary input to each neuron, which indicates whether a packet is waiting, the inputs may take on continuous values which represent the priority of the corresponding packets. Brown [103] argues that Winner-Take-All (WTA) NNs are superior to Hopfield networks in that they can achieve similar performance but do not require a pre-defined energy function and require less connectivity. The problem associated with all these techniques however is that they can only achieve the necessary speeds when implemented in hardware or software implemented on parallel processors.

A similar task of optimisation arises in the field of routing. In [104], a Hopfield network is used to carry out a shortest-path routing algorithm in which associated with each link is a particular cost and the chosen route is the one which minimises the overall cost. A conventional approach would involve determining the cost of each possible route and then selecting the best one, however as with the switching problem, an energy function is defined that encodes the goals and constraints and the NN acts to minimise this energy function. Once the energy function is minimised, the activation state of the NN provides the optimal route.

The different problem of adaptively choosing a set of optimal control parameters in a multiplexer such that the QoS requirements of the traffic are satisfied is looked at in [105]. In the learning phase, a NN learns the relationship between the network status, such as the traffic conditions and control parameter values, and the observed QoS. In the control phase a Genetic Algorithm (GA) (see [106] for an overview) generates a possible control parameter set which, along with the current traffic conditions, is used as the inputs to the NN. The NN estimates the QoS that would result from this control parameter set and this is evaluated against the desired QoS in order to obtain a fitness score. The fitness score is subsequently returned to the GA such that another control parameter set can be generated and the cycle continues. The evolutionary properties of GAs allow an optimum control parameter set to be derived.

3.3 Fuzzy Control of Networks

NN based control does not require any knowledge about how the system functions and simply relies on their learning capabilities. On the other hand, controllers based on expert systems require a detailed knowledge of the system in order to derive control rules. Fuzzy control systems

however, are able to derive control rules based on incomplete or imprecise knowledge of the system and the uncertainty associated with communication networks has lead to various applications of fuzzy control systems in this field.

Unlike traditional set theory in which an element is either a member of a set or not, fuzzy set theory allows elements to have a degree of membership with infinitely many sets. The basic idea behind a fuzzy controller is to use expert knowledge and experience to derive linguistic control rules which are typically expressed in the form of if-then statements, this can be considered as an emulation of human behaviour. The deduction of this rule is called inference and requires the definition of a membership function which allows us to determine the degree of truth associated with each proposition. A generic fuzzy controller is given in Figure 3.5.

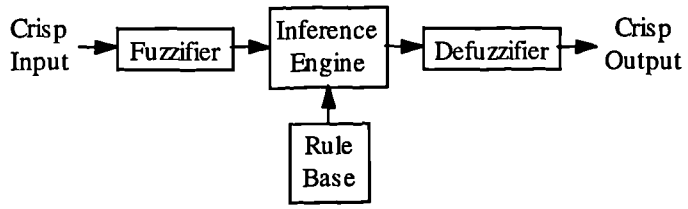

Figure 3.5:A typical fuzzy controller

The first step is known as fuzzification where a precise value is mapped onto a fuzzy variable using a membership function, this fuzzy variable represents the extent to which the value belongs to each fuzzy set. Using the linguistic control rules in the rule base, the degree of truth associated with each rule is derived and by combining these a precise output is obtained which can be used for control. The procedure for obtaining a precise output is known as defuzzification. For example, if we consider a variable BUFFER LEVEL then the fuzzy sets *empty, medium* and *full* could be used to describe the status of the buffer. A typical control rule in this situation could be 'If BUFFER LEVEL is *full* then *decrease* SOURCE RATE'.

A thorough review of fuzzy control systems can be found in [107], however the advantages often associated with fuzzy control systems include the ability to provide a robust mathematical framework for dealing with linguistic and imprecise information. Moreover, they exhibit a soft behaviour, therefore they have a greater ability to adapt to dynamic environments, and their simplicity often leads to low computational complexity. On the other hand, the problems associated with fuzzy control systems include the lack of on-line learning together with no clear and general technique for mapping expert knowledge onto the parameters associated with the

controller. Although they have not received the same attention as NNs, fuzzy control systems have been applied to network control problems such as rate control, admission control, traffic policing and buffer management.

3.3.1 Fuzzy Rate Control

The rate controller proposed in [108] is a fuzzy implementation of the two-threshold control method in which two threshold values are used to determine the onset and relief of congestion periods. When the queue length is above the upper threshold then the buffer is congested and the sources receive a signal to reduce their transmission rate, on the other hand when the queue length is below the lower threshold then the buffer is not congested and the sources receive a signal to increase their transmission rate. The conventional approach therefore relies only upon the queue length, however in the fuzzy controller the rate of change of the queue length and the current cell loss are taken into account to indicate the occurrence of congestion. The parameters and rules of the controller are determined through many analytical results of the two-threshold method and the output indicates how much the sources should increase or decrease their transmission rates. The scheme is compared with the conventional two-threshold method and leads to 4% less cell loss, the reason for this is that the fuzzy controller uses more information and can indicate the occurrence of congestion in advance whilst also providing soft and accurate control during congestion periods. However, a more informative comparison would be to compare it with a non-fuzzy approach which makes use of the additional variables. A similar backward congestion notification scheme is proposed in [109] to control ABR traffic sources in both LAN and WAN environments. The fuzzy controller is fed with the buffer occupancy level and the rate of change of this level, and responds with the new normalised explicit rate for the sources feeding the switch. The rule base is tuned by observing the progress of a simulation based on a desire to strike a balance between the maximum throughput and minimum end-to-end delay. Compared to a non-fuzzy backward congestion notification scheme, the proposed mechanism leads to higher network utilisation and a lower end-to-end delay. The application of an adaptive fuzzy control strategy such that the scheme remains optimally tuned under all network conditions is seen as further work.

3.3.2 Fuzzy Admission Control

The authors of [110] claim that NN based admission controllers that learn the non-linear relationship between the number of connections and loss rate may be ineffective. This is because they only learn average values and actual observed loss rates may disperse from this average

because of the diversity of the traffic arrival process. The proposed fuzzy admission controller estimates the possibility distribution of packet loss and the upper bound of this distribution is used to perform admission control. The fuzzy rules for the fuzzy inference are tuned automatically by the error back-propagation learning algorithm which is based on the minimisation of energy functions. The problem of estimating loss rates in the region with no observed data is overcome by extrapolating the parameters associated with existing fuzzy sets in order to generate new fuzzy rules. Simulation results are presented for the case of only one traffic class and problems may arise in the case of more traffic classes since the number of fuzzy rules increases exponentially with the number of classes. A fuzzy implementation of the equivalent capacity admission control method proposed by Guerin et al in [63] is outlined in [108]. The fuzzy bandwidth predictor uses expert knowledge from [63] to estimate the equivalent capacity required for the new call given the peak rate, mean rate and average burst duration. Unlike the original equivalent capacity method, which only uses the estimated available capacity, the fuzzy admission controller uses this value together with the observed loss rate and an indication as to the level of congestion in the network, obtained from a fuzzy rate controller (see Section 3.3.1), to determine whether or not a call should be accepted. The scheme is shown to achieve 11% greater utilisation than the original method while still satisfying the QoS requirements of the calls. The authors claim that the reason for this is that the proposed method is a hybrid of parameter and measurement based admission control, with varying degrees of importance attached to each approach.

3.3.3 Fuzzy Traffic Policing

Traditional policing mechanisms rely on crisp threshold mechanisms to control stochastic parameters and the soft decision making properties of fuzzy systems have lead to a number of proposals for fuzzy traffic policers. In [111] the ratio of the measured mean burst length of the traffic to the declared mean burst length and the ratio of the mean rate to the declared mean rate are used as inputs to a fuzzy policer. The inputs are mapped onto fuzzy variables which indicate the degree of membership of the fuzzy sets 'comply', 'sort-of comply' and 'violate'. The decision variable is in the range [0,1], with 0 meaning 'drop cell' and 1 meaning 'accept cell'. The policer described in [112] enforces the mean arrival rate of a bursty source. With this mechanism the maximum number of cells, N_i, which are considered to be non-violating in a fixed interval is dynamically updated by means of fuzzy inference rules. The principle upon which these rules are based is that in order to guarantee transparency to a conforming source, it is necessary to assign N_i higher than the declared average providing that the source maintains non-violating behaviour in the long term. The parameters describing the behaviour of a source are the average number of

cell arrivals per window since the start of the connection and the number of cells arriving in the last window. The first gives an indication of the long-term behaviour of the source and the second gives an indication as to the current behaviour. A third input parameter is the value of N_i in the last window and the output is a fuzzy variable describing the required change in N_i. In [113], a fuzzy leaky bucket scheme is proposed in which the control variable is some additional depth associated with the token bucket. This extra depth stores 'red' tokens which cause the consuming cells to be tagged, this allows for violating cells to be tagged rather than discarded when the network is at low utilisation.

3.3.4 Fuzzy Buffer Management

In [114] it is argued that the use of fuzzy thresholds leads to superior performance compared to binary thresholds. This paper introduces the notion of selective cell blocking where a switch refuses entry to a burst of cells emanating from the source, these cells must then be re-routed and will therefore incur additional delays. With a binary threshold, if the buffer occupancy is above the threshold then all arriving cells are blocked. A conservative choice of threshold will lead to very low cell loss due to buffer overflow, however the buffer utilisation will be very low and the delays high. By contrast, a higher choice of threshold will lead to greater utilisation and lower delays at the expense of increased cell loss. The goal is to achieve a reasonable trade-off between utilisation, loss and delay, and it is demonstrated that using fuzzy thresholds, based on the notion of 'fullness' to determine the probability of blocking, leads to improved performance compared to discrete binary thresholds.

3.4 Stochastic Learning Automata

A stochastic learning automaton (SLA) can be regarded as a finite state machine. Each state has a probability associated with it and corresponds to a particular action. It operates by selecting one of these actions which is then evaluated by a random environment. The response from the environment is used by the automaton to update the action probabilities using some kind of reinforcement learning algorithm and the fundamental idea behind the use of learning automata is that over time the automata converge to action probabilities which give rise to optimal performance. An overview of the theory of learning automata is given in Appendix B and a more comprehensive review can be found in [115].

The main advantage of learning automata is their sheer simplicity and the intuitive way that they operate. For example if the chosen action was a good one then the probability associated with it

is increased and if it was a bad one then the probability is decreased. Since learning automata do not assume any prior knowledge they are useful in situations where the system to be controlled has unknown characteristics or has characteristics which vary over time. However, in order for an automaton to be effective, any time variations must be sufficiently slow to allow the automaton to converge. Unfortunately, as the number of actions increases it has been shown that the convergence time increases dramatically [115] and this characteristic may therefore limit the potential applications of automata, although recently schemes such as discretised learning automata have been proposed to speed-up convergence [116]. Instead of using a single automaton with a large number of actions to control a complex system, a more appropriate approach may be to use several automata in a distributed fashion with each automaton having a small number of actions. This approach is therefore ideal in systems which are amenable to distributed control, a typical example of such a system is a communication network.

By far the most popular application of learning automata to network control is in the field of routing, however they have also been applied to problems associated with flow control and queue management. Next we shall give a brief overview of these applications.

3.4.1 Automata Routing

Due to the dynamic nature of networks the optimal routing policy is one which is able to adapt to changing conditions. Centralised adaptive routing relies on a central routing facility to assemble global knowledge of the network in order to formulate a routing strategy which is then broadcast to the individual nodes. Such schemes have the potential for optimal routing, however they suffer from additional overhead and an increased reaction time compared to distributed control. In distributed adaptive routing, the nodes make local routing decisions which are supported by global feedback gathered in co-operation with other nodes. This form of distributed control is ideally suited to learning automata since they do not require any prior knowledge about the network topology or the carried traffic.

Learning automata have been applied to routing in both circuit switched and packet switched networks [115][117]. In the context of circuit switched networks, at each node there is an automaton associated with each possible destination. The actions of each automaton correspond to viable routes and at call set-up one of the routes is chosen probabilistically. If the call-set up request is successful then a feedback signal is generated and the probability of selecting this route is increased, on the other hand if the request is unsuccessful the corresponding probability is decreased. This approach has been shown to perform at least as well as the optimum fixed rule

strategy which simply selects routes in a fixed order, moreover in situations which require a mixed routing strategy they have been shown to be superior [115]. Furthermore, since no prior knowledge of the network is assumed, improved performance may be realised by incorporating network status information in the operation of the automata.

In packet switched networks, learning automata have been proposed for both virtual call and datagram networks. Routing in virtual call networks is very similar to circuit switched networks, however datagram networks may route the individual packets as separate entities such that the traffic is spread evenly over the available capacity in the network. At each node in a datagram network there is an automaton corresponding to every possible destination i.e. in a N node network there are N-1 automata at each node. The actions of each automaton correspond to the outgoing link a packet should be routed on in order for it to reach its destination. Several possible performance criteria exist, for example link utilisation, throughput and average hop count, however the most popular performance measure is packet delay. Therefore when a packet reaches its destination, an acknowledgement packet is returned to the source to indicate that the packet was received together with the experienced delay. These delays are then used to update the action probabilities of the appropriate learning automaton, the overall scheme is summarised in Figure 3.6. This approach has been shown to outperform shortest-path and random routing since it spreads the load over the network in an intelligent manner [117]. A more recent investigation of the use of learning automata for routing can be found in [118].

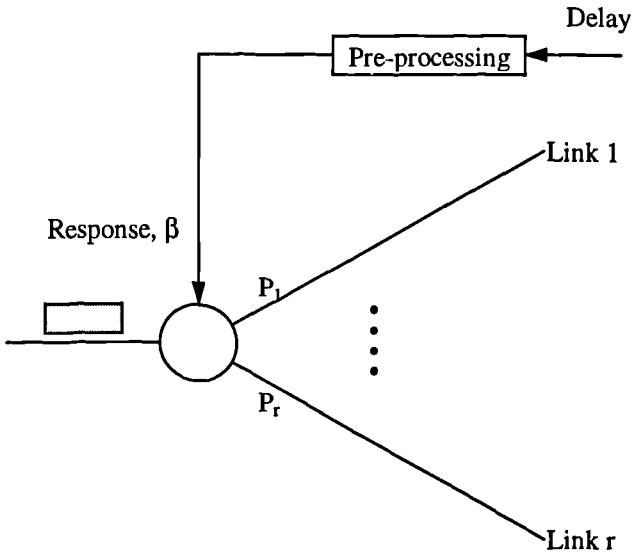

Figure 3.6:Learning automata datagram routing

3.4.2 Automata Flow Control

In this application, learning automata are used to control the rate at which messages enter a packet switched data network. This regulation avoids the situation where the load on the network is too great, hence resulting in congestion (N.B such control is not applicable to integrated services networks since a separate admission controller performs this task). In order for a message to gain entry into the network it must receive a permit which is tagged onto the message. When the message reaches its destination the permit becomes available for other messages and by limiting the total number of permits the rate of message influx is controlled. A conventional approach, in which delivered permits remain at the destination until a message requests access to the network at that node, will result in an uneven distribution of permits under asymmetric traffic conditions. Two approaches to automata based flow control are suggested in [119], namely centralised and de-centralised. In the centralised scheme, permits reaching their destination are routed to a central automaton which, based on the some performance criteria such as loop delay or loop population, updates its action probabilities which determine the likelihood of redistributing permits to particular nodes. In the de-centralised approach, an automaton is located at every node and each action corresponds to redistributing an arriving permit to a particular source node. As with the centralised case, on the arrival of a permit the action probabilities are updated based on some performance measure. It is the aim of the scheme to optimise the flow through the network and simulations were carried out to compare the centralised automaton scheme using various performance measures to the optimal. It was found that the performance using the various feedback measures was dependent upon the number of permits, network topology and traffic conditions, although generally all schemes were close to the optimal.

3.4.3 Automata Queue Management

In [120], learning automata are applied to the problem of priority assignment in queueing systems. The authors consider two classes of jobs which have Poisson arrival rates and exponential service times, and within each class the service is FIFO. Providing that the mean arrival and service rates are known it is possible to determine the optimal service strategy which minimises the overall delay. It is demonstrated that a learning automaton whose action probabilities correspond to selecting a particular queue is able to approach the optimal fixed rule without any prior knowledge of the arrival and service rates. We feel that the ability of learning automata to adapt to give near optimal performance in queueing systems with unknown parameters has not been exploited and this application is investigated later in the thesis.

3.5 Summary

In this chapter we have presented various AI techniques and critically reviewed their applications to network control. We identify three basic uses for NNs when applied to the problem of network control. Firstly, they can been used to approximate complex relationships in admission control, these relationships are either unknown or they take considerably more computational effort using conventional approaches. Secondly, they can been applied to the problem of optimisation in switching, routing and link allocation, however the advantage of NNs in this application is their parallel processing properties which can only be utilised when implemented in hardware or implemented in software using parallel processors. Finally, they can be used in flow control and traffic policing to perform traffic prediction, in such situations the NNs learn the relationship between past and future arrivals.

The use of fuzzy logic has also received considerable interest, however, there is a major problem associated with these techniques in that it is difficult to map the expert knowledge to the fuzzy parameters. The only advantage we can identify with these schemes over conventional approaches is that they require less computational effort.

We have also looked at the application of learning automata for routing and flow control, and it appears that learning automata are ideally suited to these tasks since they are amenable to distributed control. Also, the adaptive nature of learning automata allows them to optimally control queueing systems when the queue parameters are unknown. In the application discussed, the goal was to minimise the overall mean queueing delay, however in future integrated services networks this will not be the goal and we feel that there is scope for the use of learning automata to provide other performance guarantees.

Chapter 4

Neural Network Traffic Prediction

It is generally accepted that it is possible to make predictions about network traffic on time scales of the order of hours, days and weeks [121]. For example it is possible to identify daily trends since some working days are typically busier than others and weekends are generally quiet. Such predictions are based on the average aggregate behaviour of many network users and the law of large numbers allows for accurate prediction. However, the task becomes much more difficult when we consider shorter time scales such as the frame rate of coded video. Accurate prediction of traffic at these time scales would be beneficial in tasks such as traffic smoothing [122], dynamic bandwidth allocation [123] and flow control [99]. It has been demonstrated that neural networks are capable of learning complex non-linear relationships [124] and recently this capability has been applied to the problem of traffic prediction [125][126][127][128][129][130]. Authors in this area claim that neural networks are able to learn the correlations and regularities that exist in traffic arrivals and hence capture the unknown complex relationship between past and future arrivals. In this chapter we investigate the suitability of such schemes and apply them to a variety of traffic types. We argue that the reported successes are valid only in certain special cases and that, in general, accurate prediction is not possible. This argument seems reasonable from an intuitive standpoint since the features of a video traffic stream are governed by scene content, and it is impossible to predict future events in live video broadcasts, such as sport. The remainder of this chapter is organised as follows. First we shall present some motivation and outline various schemes that would benefit from accurate traffic prediction. Next we define the problem as a standard time series prediction task and identify methods of measuring accuracy. Simulation results are then presented for a variety of traffic types and we explain why some traffic types are easier to predict than others. Finally we summarise the chapter and present some conclusions.

4.1 Motivation

As one would expect an ability to see into the future would be a great asset to any network control function. In this section we identify three tasks that would benefit from an accurate traffic prediction scheme ; traffic smoothing, dynamic resource allocation and flow control.

4.1.1 Traffic smoothing

Compressed video typically exhibits significant burstiness on multiple time scales due to the frame structure of the compression algorithm together with scene variations [131]. Video smoothing is a natural approach to reducing the bandwidth variability of video. It relies upon an ability to determine smooth transmission rates such that the client buffer never overflows or underflows (the buffer underflows when a frame fails to arrive before its playback time). It has been shown that video smoothing can improve network utilisation significantly [132]. Clearly an ability to predict future video frame sizes would be advantageous since this would allow for a more accurate calculation of the required transmission rate, an example of such a scheme is presented in [122].

4.1.2 Flow control

A flow control scheme using neural network traffic prediction is proposed in [99]. The main control action of such a scheme is to reduce the peak rate of the traffic sources when the neural network predicts possible congestion in a multiplexer. Based on the current buffer occupancy and a prediction of the number of arrivals in the next time window, the controller predicts the future buffer occupancy. If the predicted buffer level is above a certain threshold, a feedback signal is sent to the traffic sources requesting a reduction in the arrival rate, this can be achieved by increasing the coding rate. The advantages of such a scheme include that it is not of the reactive control type and any control action taken will be in time to alleviate potential congestion

4.1.3 Dynamic Resource Allocation

The authors of [123] propose a dynamic bandwidth allocation scheme based on traffic predictions. The neural network predicts the bandwidth requirement of each incoming traffic stream for the next time window. If the sum of the predicted bandwidth is larger than the link capacity then the bandwidth allocated to each stream is reduced accordingly. Conversely, if the sum of the predicted bandwidth is less than the link capacity the excess bandwidth is distributed evenly between the streams.

The possible applications of an accurate traffic prediction scheme are limitless, for example it may be used to dynamically set thresholds in the partial buffer sharing scheme described in the Chapter 2. Such applications will not require an accurate characterisation of the traffic in order to select control parameters, they would simply select the control parameters based on predictions.

4.2 Time Series Prediction

The problem of traffic prediction is a standard time series prediction task, the goal of which is to forecast the value of a variable based on previous observations of that variable. The scalar time series is denoted by $x(n)$, and it is assumed that there is some function f such that,

$$x(n) = f\,(x(n-1),\, x(n-2),\, x(n-3),\ldots..,\, x(n-q)) + \varepsilon(n) \tag{5.1}$$

where q is the order of the function and $\varepsilon(n)$ is a residual which is assumed to be white noise. This function is unknown, and the only information available is the set of observables : $x(1)$, $x(2),\ldots,x(N)$, where N is the total length of the time series. It is the goal of the prediction scheme to approximate this function, and estimate $x(n)$ given the previous p observations, as shown by

$$\hat{x}(n) = F(x(n-1), x(n-2), x(n-3),\ldots..,x(n-p)) \tag{5.2}$$

where F is an approximation of the unknown function f.

Neural networks have been successfully applied to the problem of time series prediction [133]. They are capable of learning complex non-linear relationships, and it has been proved that a three-layer feedforward neural network, with sigmoidal units in the hidden layer and trained using the back propagation algorithm, is able to approximate an arbitrary non-linear function [124].

4.3 Performance Measures

Several methods exist for determining the accuracy of time series predictions, some more suitable than others. The goal of this section is to determine a fair method by which to quantitatively compare various schemes applied to different time series. We must stress that although it is impossible to quantify, valuable information about the accuracy of a prediction can also be obtained by simple inspection.

4.3.1 Statistical Properties

Various statistical properties of the predicted and actual data may be compared, these include the marginal distribution and autocorrelation function [129]. Although such statistical properties may compare favourably, the only conclusion that one may draw is that the prediction is statistically similar to the actual data. Unless there is a direct comparison of the predicted and actual data, claims that a particular scheme is successful are inconclusive. For example, consider a scheme that predicts half the results of a coin tossing experiment to be heads. The predicted data will, on the whole, have the same probability distribution as the real data, however, it is still possible for the scheme to predict the wrong result every time. Furthermore, a simple prediction scheme that predicts the next value to be the same as the current value would possess almost identical statistical properties to the actual data set since they are essentially the same sequence shifted one step in time.

4.3.2 Mean Squared Error

Another widely adopted performance measure is the mean squared error (MSE) [99][126].

$$MSE = \frac{1}{n}\sum_{i=1}^{n}\left(x_i - \hat{x}_i\right)^2 \tag{5.3}$$

Such a method allows for direct quantitative comparison of predictions on the same data set, however the lack of any normalisation does not give any real meaning to the calculated value. Hence a problem occurs in deciding what value of MSE gives satisfactory performance for a particular data set. Furthermore, the lack of any normalisation gives it limited use when comparisons of different data sets are required. For example, the dotted lines in Figure 4.1 represent trivial predictions of the corresponding data sets where the predictor simply predicts the mean.

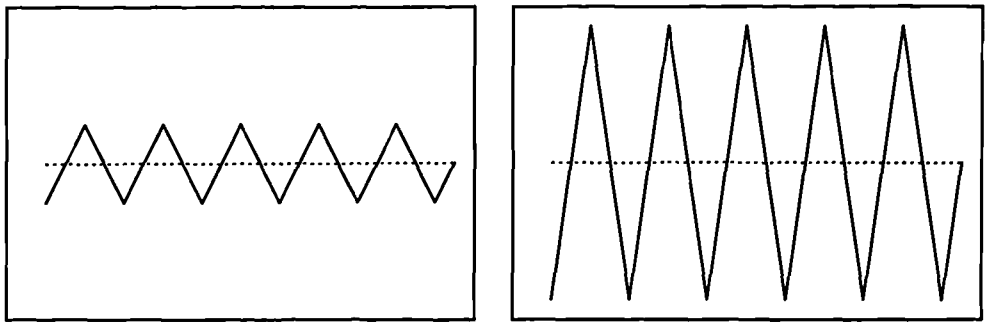

Figure 4.1:Trivial prediction of different data sets

Clearly neither predictor can be considered superior to the other, however the leftmost predictor results in a smaller MSE due to the smaller variance of the original data set. This limitation is overcome by normalising the MSE with the variance of the actual data, the resulting value is known as the coefficient of determination.

4.3.3 Coefficient of Determination

The coefficient of determination, r^2, is a function of the mean squared error normalised by the variance of the actual data.

$$r^2 = 1 - \frac{\sum_{i=1}^{n}(x_i - \hat{x}_i)^2}{\sum_{i=1}^{n}(x_i - \bar{x})^2} \tag{5.4}$$

For a perfect predictor the coefficient of determination is one, whereas for a trivial predictor that simply predicts the mean of the data the coefficient of determination is zero. This measure thus allows an unbiased comparison of prediction schemes applied to different data sets.

4.4 Traffic Prediction Techniques

In the remainder of this chapter we apply a variety of time series prediction techniques, based on both neural network and linear regression approaches to a number of traffic types. First we shall outline the techniques adopted.

4.4.1 Feedforward NN Trained Using Back-Propagation

A feedforward neural network trained using the back-propagation algorithm is the most common approach adopted in the literature [125][126][128][129], a review of neural networks can be found in Appendix A. Typically a 3-layer network is used, however the number of neurons in each layer varies. The most common application of traffic prediction is in the prediction of VBR video traffic, thus we shall describe a typical approach for this scenario. A scheme with 5 input neurons, 5 hidden neurons and a single output neuron is shown in Figure 4.2. We shall denote such an architecture as 5-5-1, however it must be stressed that this is just an example architecture and the optimal architecture can only be found through trail and error. The inputs to the neural network are the previous frame sizes and the output is a prediction of the size of the next frame. The blocks labelled z^{-1} represent one step delay elements.

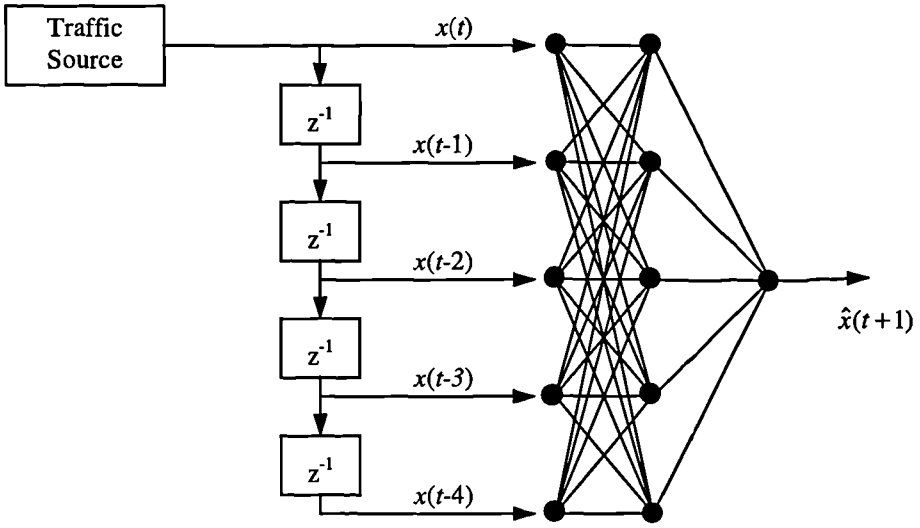

Figure 4.2:A 5-5-1 neural network for traffic prediction

Two different approaches to training the neural network are adopted, an off-line approach and a much more computationally expensive on-line approach.

4.4.1.1 Off-Line Training

In this approach, the neural network is trained before being exposed to a real network situation. A training set consisting of example traffic is presented to the neural network and after each training example the weights are updated according to the back-propagation algorithm. After a complete pass through the training set, the weights are frozen and the neural network is tested on another data set, known as the test set. The performance on this test set determines how well it can generalise from the examples presented to it during training. Therefore the test set mimics a real network situation in which the neural network is exposed to previously unseen data and the accuracy of the prediction scheme is judged using this data. The advantage of this scheme is that it is extremely fast (after the initial training phase) which makes it applicable to real-time forecasting. The drawback however is that this method assumes that the traffic is stationary and maintains the same characteristics over time.

4.4.1.2 On-Line Training

In contrast to the off-line training method, in this approach the neural network is continually learning. After an initial off-line training phase the training is performed on a data set that characterises the most recent arrival pattern and a prediction of the next frame is made. Whenever a new frame becomes available the training set is shifted to include this frame and the

oldest frame is discarded. The training scheme is outlined in Figure 4.3.

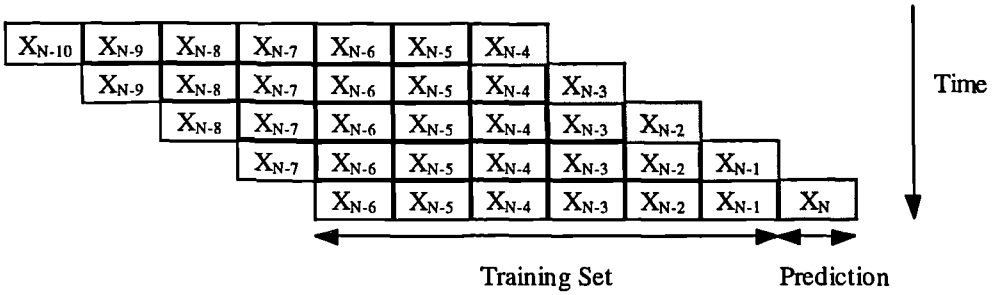

Figure 4.3:On-line training method

After the training set is shifted, the new training session starts from the existing weights rather than from random values, therefore it is reasonable to expect convergence in a significantly smaller number of iterations compared to off-line learning. The advantage of on-line training is that the continuous learning on new examples should cope with changes in the distribution and characteristics of the sequence. However, the additional training adds to the computational complexity and thus this approach may not be feasible for real-time forecasting, especially if for accuracy reasons the training set or the number of iterations must be large.

4.4.2 Speed Of Convergence

In the subsequent experiments we emphasise that the goal is to determine whether accurate prediction is possible and not to investigate the various ways in which the learning process can be accelerated. Methods of increasing the learning rate include the addition of a momentum term to the weight update equation or the use of an asymmetric activation function. Unless otherwise indicated we shall assume a learning rate of 0.1, a training and test set of size 200 and a logistic activation function given in (5.5), where v_j is the net internal activity of neuron j, and y_j is the output of the neuron.

$$y_j = \frac{1}{1+\exp(-v_j)} \tag{5.5}$$

Training is carried out until it is clear from the progress of the coefficient of determination that further training would be fruitless and that the neural network performance has been optimised. Thus we attempt to demonstrate the upper limits of achievable performance.

4.5 Prediction of Model Generated Traffic

4.5.1 Video Teleconference Traffic

Early work in the field of traffic prediction [123][125][126] focused on predicting traffic generated using models, mainly due to the lack of real traffic traces. The most popular model was based on a first order autoregressive Markov process that models the correlations observed in real teleconference traffic [23].

The traffic generation process is given by,

$$X(n) = aX(n-1) + bW(n) \tag{5.6}$$

where $X(n)$ is the bit rate during the nth frame, $W(n)$ is a Gaussian random variable and a and b are constants. In [23] it is shown that the most appropriate values for the parameters are a=0.8781, b=0.1108 and the mean and variance of $W(n)$ are 0.572 and unity respectively.

Figure 4.4 shows the performance of a 1-3-1 network on some unseen data generated using the model in (5.6). The performance of the neural network stabilised after about 300 iterations with a training set of 200 examples.

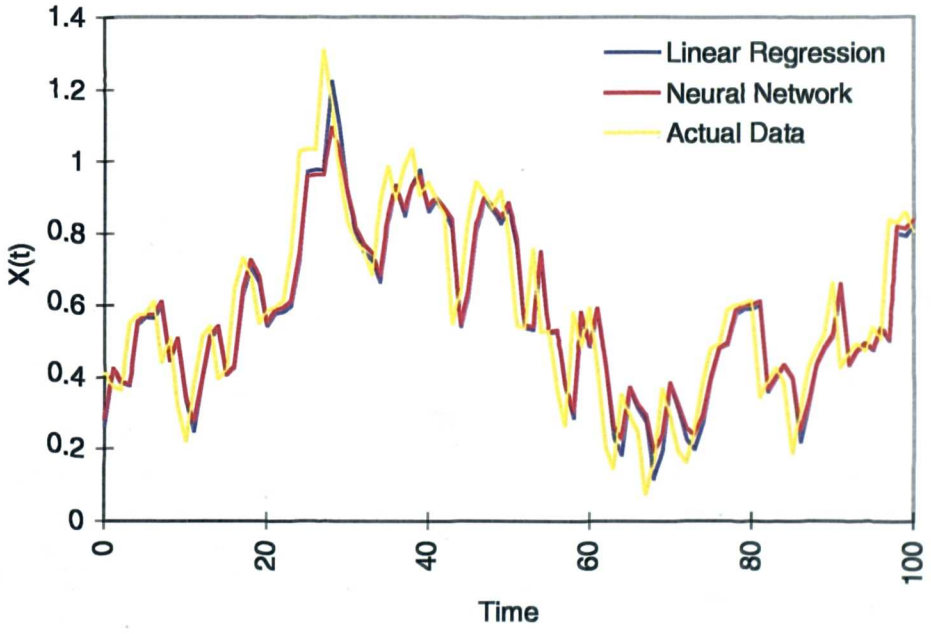

Figure 4.4:1-3-1 neural network (r^2= 0.72) and 1st order linear regression (r^2= 0.73) prediction results of model generated teleconference traffic

Neural networks with more neurons in the input and hidden layers produced very similar performance. The authors of [123][125][126] claim that the prediction in Figure 4.4 is accurate, however the results are viewed on a much longer time scale thus the predicted and actual data sets have a tendency to appear very close. Inspection of the data at a higher resolution as in Figure 4.4 clearly shows that the predictor is almost tracking the actual data, and simply predicting the next value to be same as the current value. Therefore the predicted sequence will be statistically similar to the actual sequence, a characteristic which the authors of [123][125][126] claim proves successful prediction.

The reason for this tracking can be explained by considering the traffic generation process. The size of the next frame is proportional to the current frame plus some random noise. Substituting the values of a, b, and the mean of the noise term into (5.6) yields,

$$X(n) = 0.8781\,X(n-1) + 0.063 \tag{5.7}$$

Since the frame sizes are of the order of unity, it is clear that if we assume that the Gaussian random variable is equal to the mean of the distribution, the size of the next frame will be approximately the same as the current frame. Effectively the neural network learns the relationship in (5.7), since it cannot possibly predict the noise term and the best it can do is predict the mean. However, one must raise the question of whether a neural network is in fact necessary and whether a more simple scheme would suffice. Since the traffic is generated by a first order linear model, we applied a first order linear regression fit to the training data and tested its performance on the test data, the results are shown in Figure 4.4. Such a fit generated the model given in (5.8) and clearly the traffic generation process given in (5.7) is within the 95% confidence intervals of (5.8). Moreover a coefficient of determination of 0.73 was achieved which is in fact slightly better than the 0.72 achieved using the neural network.

$$X(n) = (0.893 \pm 0.063)\,X(n-1) + (0.053 \pm 0.033) \tag{5.8}$$

4.5.2 Fractal Traffic

It is often assumed that a linear regression model is only suitable when the traffic is generated using a linear traffic model. However, assuming only linear traffic models is naive and any prediction scheme must be tested on traffic which is non-linear. In these situations non-linear regression techniques may be applied, however, their complexity may limit practicality. The

capability of neural networks in performing non-linear mappings leads us to believe that in such circumstances a neural network should easily outperform a linear prediction.

A significant amount of work has been done on the analysis of real network traffic and several authors have concluded that real traffic is self-similar in nature [27]. To capture this fractal behaviour, the authors of [27] proposed to model the traffic using a chaotic time series. The chaotic time series is generated using a simple non-linear deterministic equation, called the logistic map.

$$X_n = 4X_{n-1}(1 - X_{n-1}) \tag{5.9}$$

The performance of a first order linear regression and a 1-3-1 neural network (after 700 iterations) on a test set generated using the model in (5.9) is shown in Figure 4.5. As expected the neural network performs much better than the linear regression, the coefficients of determination are 0.99 and -0.02 respectively.

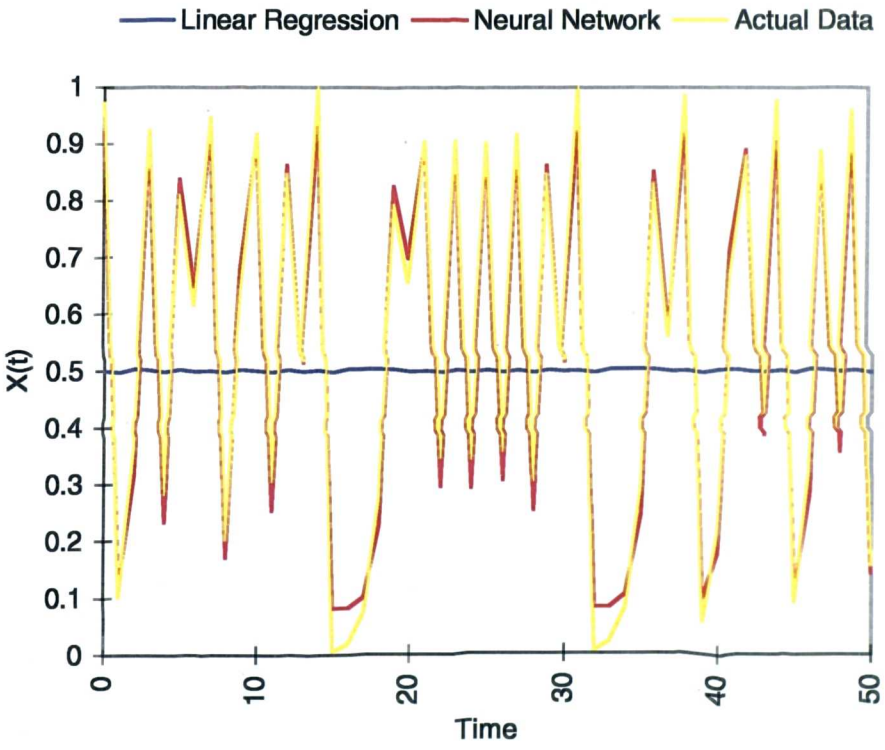

Figure 4.5:1-3-1 Neural network (r^2=0.99) and linear regression (r^2=-0.02) prediction of fractal traffic

The success of the neural network scheme leads the authors of [126] to conclude that traffic

prediction is possible. However, once again we must consider the traffic generation process before any firm conclusions can be made. From (5.9), it is clear that the traffic generation process, although chaotic, is very definitely deterministic. Therefore it is not surprising that the neural network is able to learn this non-linear mapping and the linear regression technique is not. Thus, a statement that neural networks can predict real network traffic based on these results assumes that real network is governed by a deterministic process and that there exists a relationship between past and future arrivals.

4.5.3 Markov Modulated Deterministic Traffic

In [134] the authors use a neural network to predict voice traffic which enables efficient smoothing. The traffic is generated using a Markov Modulated Bernoulli Process (MMBP). Such a process alternates between an idle state, in which no packets are generated, and an active state, in which packets are generated according to a Bernoulli process. The periods in both states are assumed to be geometrically distributed. The authors claim that the neural network can predict the duration of an active state and also the number of packets generated. In this work there are no direct comparisons between the predicted values and actual values, only a comparison of the performance of a voice smoother using this prediction technique with other more primitive traffic smoothers. The authors appear to contradict themselves somewhat since they claim that the neural network can predict the duration of the active states, however the traffic model assumes that these are geometrically distributed. Hence it appears that the neural network can predict random numbers, which surely is impossible. To prove this we applied the neural network architecture proposed in [134] to a simpler problem in which the goal is to simply predict the length of active periods in a Markov Modulated Deterministic Process (MMDP). A MMDP differs from a MMBP in that while in the active state packets are generated at a constant rate rather than according to a Bernoulli process. The prediction result is given in Figure 4.6 and as expected the neural network is unable to predict the random burst lengths and simply learns that the best performance is achieved by predicting the mean, which in this particular case is 10.

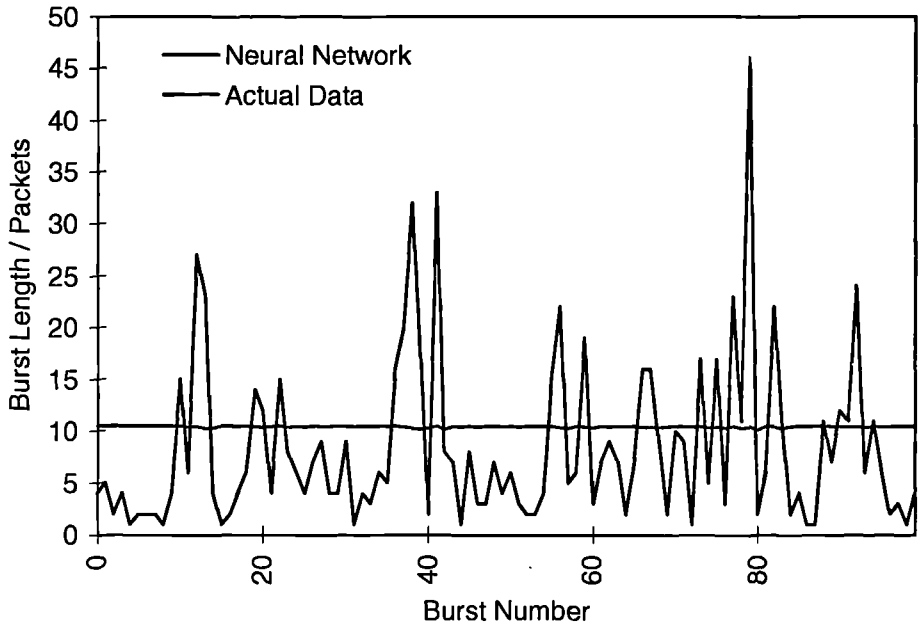

Figure 4.6:3-3-1 neural network prediction (r^2=-0.03) of MMDP traffic

In this section we have investigated three types of model generated traffic and for each type it has been reported in the literature that neural network prediction is indeed possible. We conclude however that the accuracy of any prediction depends only on the traffic generation process. For a purely deterministic generation process, as with the fractal traffic, accurate prediction is possible. However the prediction becomes more difficult as the degree of randomness increases, this is evident from the reasonable performance on the partially deterministic teleconference model and the very poor performance on the purely stochastic MMDP model. Therefore, if prediction of real traffic is indeed possible then the traffic generation process must follow a deterministic process with possibly a degree of superimposed randomness and in the following section we investigate if this is the case by performing predictions on real network traffic.

4.6 Prediction of Real Video Traffic

4.6.1 MPEG-I Video Conference Traffic

The trace we consider in this section is a MPEG-I encoded stream taken from a video teleconference session [135]. The scene content is typically head and shoulders with very little zooming or panning, further analysis of the data set can be found in [136]. Using a number of different neural network architectures, predictions were carried out using this trace and it was found that a 12-3-1 network gave the best performance. Generally, networks with 12 or more neurons in the input layer performed much better than those with less than 12. The reason for

this is that a group of pictures in MPEG-I encoding consists of 12 frames, this means that every 12th frame is coded in the same way. Further details of the MPEG-I coding algorithm can be found in Appendix C.

Linear regression fits of differing orders were also made to the data and optimal performance was achieved using 12th and higher orders. Figure 4.7 compares the performance of the 12-3-1 neural network (after 200 iterations) and a 12th order linear regression fit on the test data, the coefficients of determination were 0.96 and 0.97 respectively.

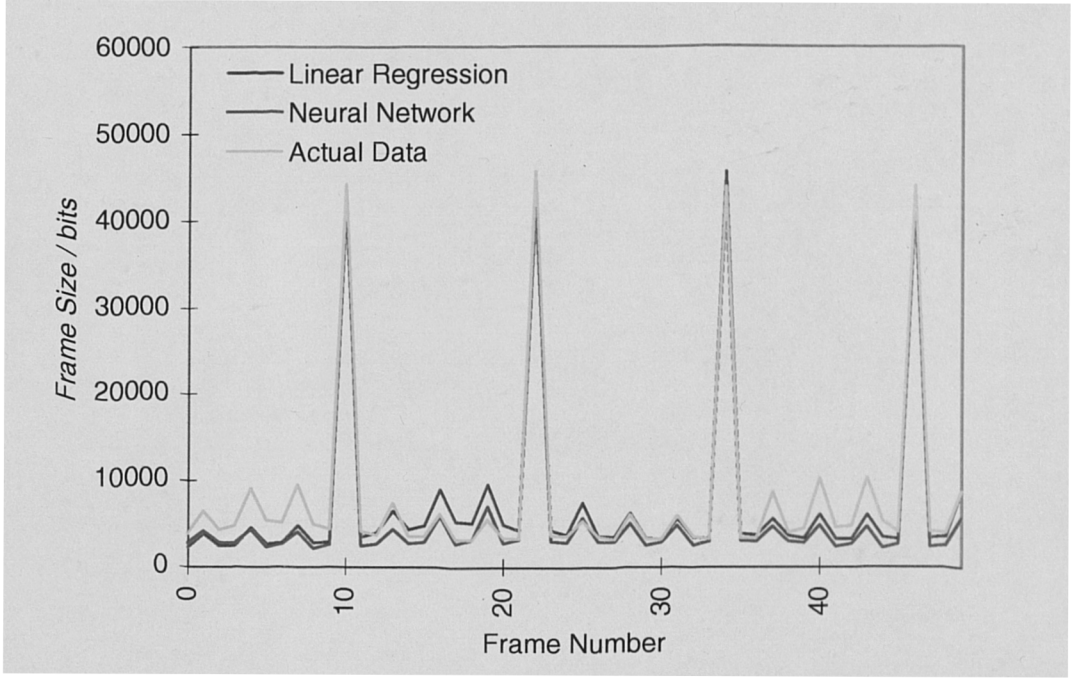

Figure 4.7:12-3-1 neural network (r^2=0.96) and 12th order linear regression (r^2=0.97) prediction of teleconference traffic

Clearly the performance of both approaches is very similar. In effect, both schemes tend to predict the next group of pictures to be the same as the current group of pictures, therefore the predictors obey the relationship,

$$\hat{X}(n) = X(n-12) \tag{5.10}$$

Through observation of Figure 4.7 it is clear that the video sequence is fairly repetitive, the reason for this being the low activity of the scenes. Thus one can conclude that the repeating nature of MPEG-I teleconference traffic makes it relatively straightforward to predict and a neural network approach holds no advantages over simple linear regression. A more challenging prediction task is that of predicting frame sizes in a video sequence of high activity, next we

investigate a movie sequence.

4.6.2 Movie Video Traffic

In this section we consider both the MPEG-I and JPEG frame size sequences of Star Wars, a typical movie with varying levels of activity [137].

4.6.2.1 MPEG-I Movie Traffic

This video sequence differs from the teleconference sequence in that although it has the typical MPEG-I structure, the diversity of scenes make it far less repetitive. This is apparent from the training set shown in Figure 4.8, and clearly some of the P-frames are as large as the I-frames (See Appendix C for terminology).

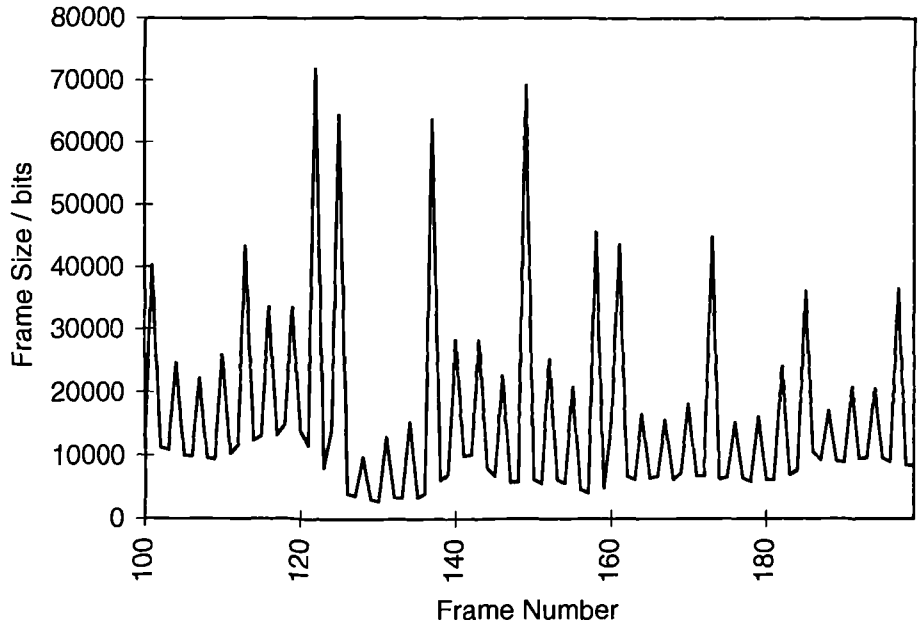

Figure 4.8:MPEG-I Star Wars training set

As with the teleconference sequence a noticeable difference in performance was observed when 12 or more input neurons were used and the number of hidden layer neurons had little effect. The prediction results for a 12-3-1 neural network (after 2000 iterations) and a 12^{th} order linear regression prediction are given in Figure 4.9. Reasonable neural network prediction is possible, however once again a linear regression fit of the same order yields similar performance. In the same way as with the teleconference sequence, both prediction schemes tend to predict the next group of pictures to be the same size as the previous group of pictures, this is evident from the large discrepancy of both predictors when the I-frame size doubles at about frame number sixty. Therefore one can conclude that due to the structured nature of MPEG-I it is possible to achieve

reasonable prediction, however as the scene activity increases the prediction becomes more difficult.

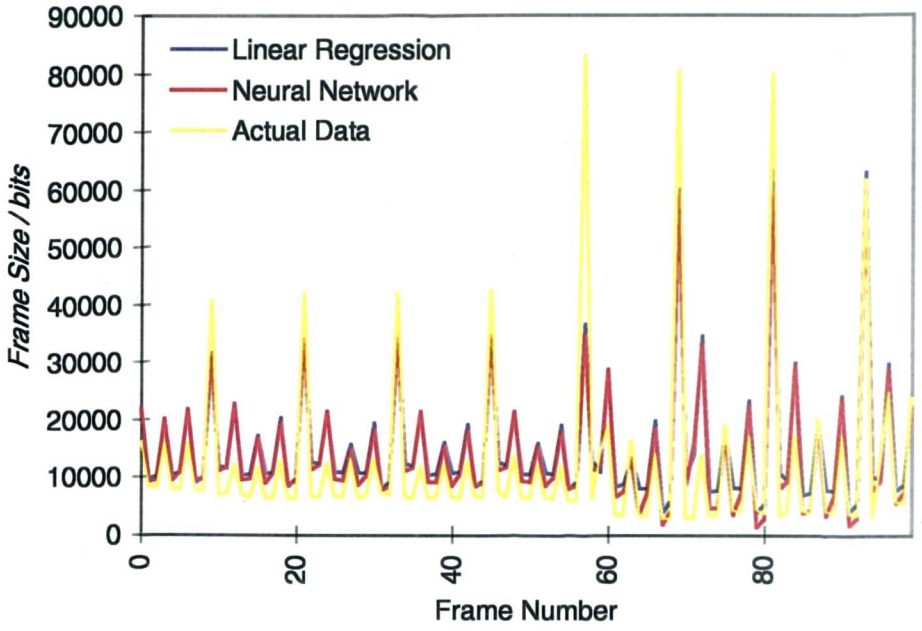

Figure 4.9:12-3-1 neural network (r^2=0.76) and 12th order linear regression (r^2=0.76) prediction of Star Wars

If frames of the same type are treated separately then the sequences generated would not possess the highly structured nature of conventional MPEG-I. An attempt to predict the I-, P- and B-frame sizes based on the size of previous frames of the same type is carried out in [130]. This approach differs from many others in that the authors use a recurrent neural network instead of a feedforward network, the reason for this choice being an increased learning rate. Once again the authors claim successful prediction is possible, however as with several other cases the authors compare statistical properties, which we have shown can give misleading results, and view the traces at low resolution, therefore the predicted and actual data appear very close. When the traces are viewed more closely they indicate that the prediction simply tracks the actual data. The prediction of a sequence of I-frames from the Star Wars trace is given in Figure 4.10 using a 3-3-1 feedforward neural network with back-propagation, clearly similar results to using the recurrent network in [130] are obtained with the prediction lagging the actual data by one frame. Also shown in Figure 4.10 is a 3rd order linear regression fit. Clearly the performance is comparable to the neural network.

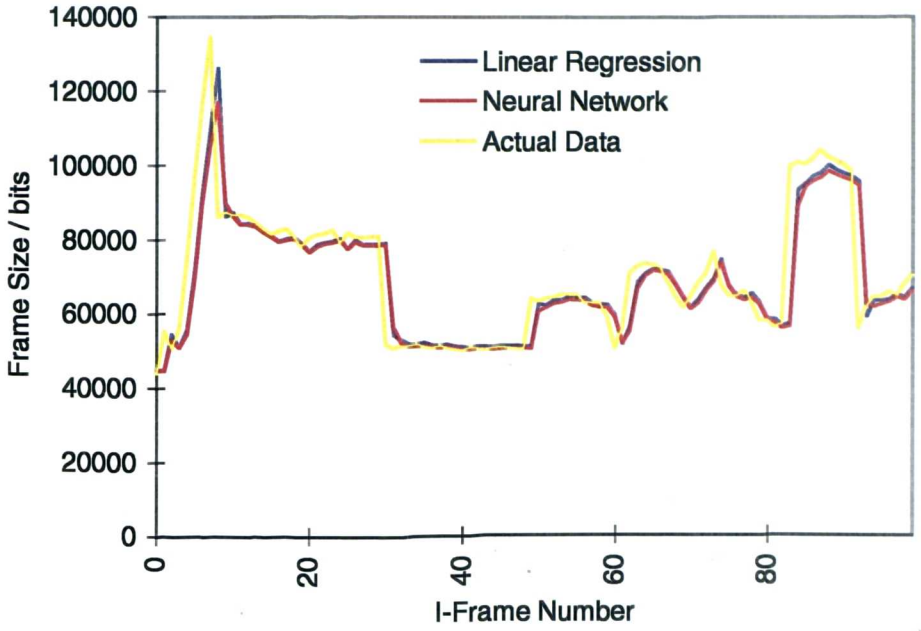

Figure 4.10:3-3-1 neural network (r^2=0.69) and 3rd order linear regression (r^2=0.69) prediction of I-frame sequence from Star Wars

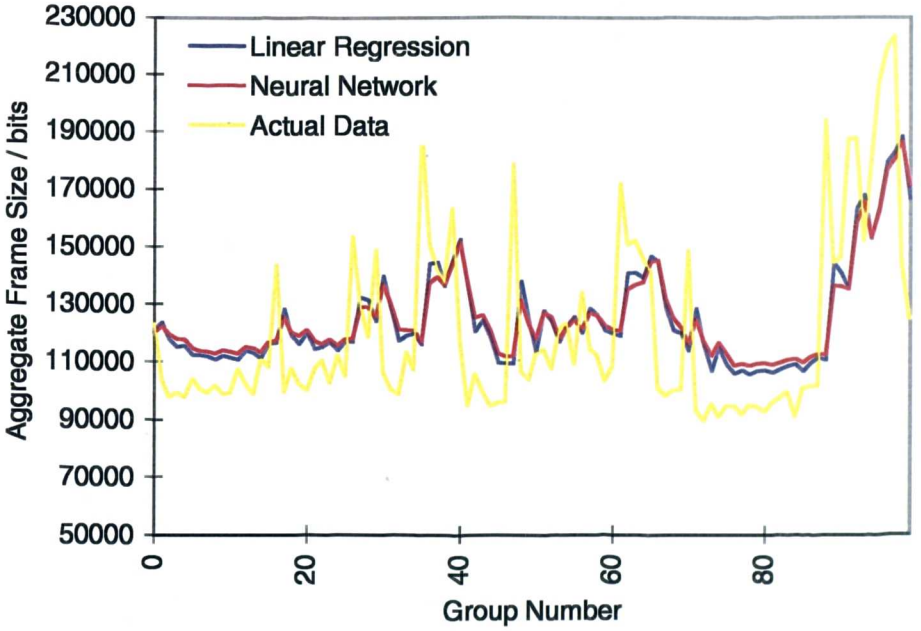

Figure 4.11:5-3-1 neural network (r^2=0.34) and 5th order linear regression (r^2=0.36) prediction of the aggregate frame size of a group of pictures from Star Wars

Another way to remove the structure of the MPEG-I trace is to consider the aggregate bit rate within a group of pictures, thus the sequence generated represents the sum of 12 frames which corresponds to about half a second of video. Experiments were conducted on this data set for various neural network architectures, and although the results for each were quite similar the

optimal architecture in terms of accuracy and complexity was a 5-3-1 network. The corresponding prediction performance for a neural network and a linear regression fit of the same order is given in Figure 4.11. The prediction is relatively poor, once again with the neural network having a tendency to track the actual data. However, in this case the neural network is quite slow to respond to jumps in the actual data. This is because jumps in the bit rate of the training set are short lived and more accurate results can be achieved if the predictor responds slowly and generally remains close to the mean. Similar performance is achieved with the linear regression predictor.

4.6.2.2 JPEG Movie Traffic

This sequence is coded differently to the MPEG-I sequence investigated in the previous section in that only spatial redundancies are reduced and each frame is coded in the same way. Figure 4.12 shows the prediction of this trace using a 5-3-1 network and a 5th order linear regression fit. As before, both schemes perform similarly and the predictions simply follow the actual data. This differs from the aggregate MPEG-I trace in that the predictor is faster to respond to bit rate fluctuations, this is because in this trace jumps are often followed by a stable period.

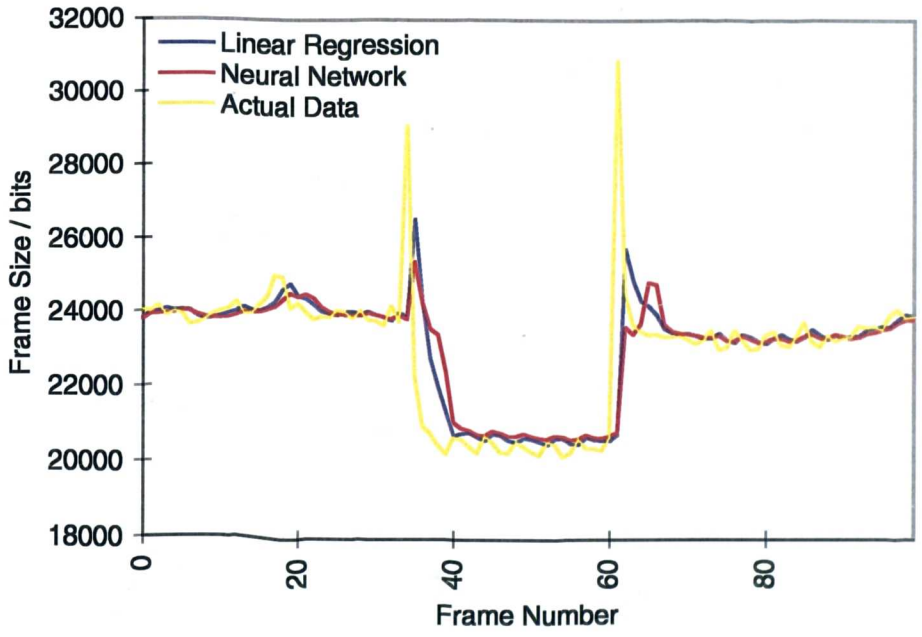

Figure 4.12:5-3-1 neural network (r^2=0.42) and 5th order linear regression (r^2=0.43)prediction of JPEG Star Wars Trace

In this section we have studied various video traffic traces. We have found that the periodic nature of MPEG-I traffic allows for reasonable prediction using both neural networks and linear regression techniques, however prediction becomes increasingly more difficult as the activity of

the video sequence increases. For video sequences not possessing a definite structure, prediction was difficult and the best performance was achieved by simply tracking the original trace. The degree with which the predictor tracks the original trace was dependent on the trace. For traces with a rapidly fluctuating bit rate the predictor was slow to respond to jumps in the bit rate and had a tendency to remain close to the mean, an example of such a trace was the aggregate MPEG-I trace. Alternatively, for traces such as the I-frame or JPEG sequence that consisted of jumps in the bit rate followed by stable periods, the predictor responded much more rapidly.

4.7 Prediction of Data Network Traffic

This data set contains the amount of traffic transmitted per 10ms interval over a 10Mb/s Ethernet [138], further analysis of the trace can be found in [27]. The optimal neural network architecture was found to be a 5-3-1 network and the corresponding performance is given in Figure 4.13 together with a linear regression fit of the same order.

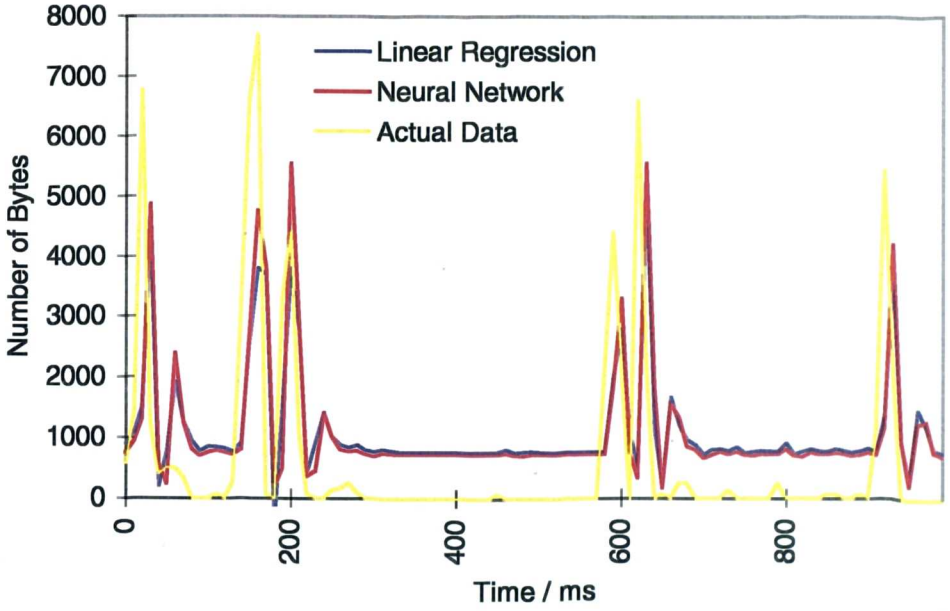

Figure 4.13:5-3-1 neural network (r^2=0.12) and 5th order linear regression (r^2=0.16) prediction of Ethernet trace

The prediction is poor and although peaks occur at roughly the same times, in each case the predicted peak occurs after the actual peak, and on no occasion does a prediction pre-empt an actual peak. Moreover, the prediction is consistently above the target value when the target is zero, this occurs because both schemes learn that the best performance is achieved when the mean of training sequence is predicted.

The authors of [128] claim that a 30-30-1 neural network is capable of predicting a superposition of the MPEG-I Star Wars trace and Ethernet trace. However, close inspection of the traces reveals that the video contributes most of the aggregate traffic, thus the superposition will retain much of the MPEG-I structure which is relatively easy to predict. Simulations were carried out using this network architecture and as expected the performance was no better than the results already presented.

4.8 Alternative Prediction Techniques

In the previous investigations we have concentrated on the use of a standard feedforward network trained off-line using the back-propagation algorithm. In this section we discuss the consequences of applying an Finite Impulse Response (FIR) neural network to the problem and also performing training on-line.

4.8.1 FIR Neural Network

One disadvantage of the standard back-propagation algorithm is that it can only learn a mapping which is static and independent of time. This is fine for tasks such as pattern recognition in which the mapping is purely spatial, however time is most definitely a factor in network traffic prediction and thus it may be advantageous to adapt the standard feedforward neural network such that it can respond to time varying signals. One approach is the time delay neural network (TDNN), here the outputs of each layer are buffered several time steps and then fed fully connected to the next layer. Such a neural network has already been applied to problems such as speech recognition [139]. A TDNN is in fact equivalent to a standard feedforward neural network where each synapse is represented by a finite impulse response (FIR) filter (See Appendix A). An FIR network may be trained by unfolding the network in time and using standard back-propagation, however a more efficient but approximate method is the temporal back-propagation algorithm [140]. It is of interest to note that an FIR network won the Santa Fe Institute Time Series Prediction Competition [141] from a diverse list of submissions, including standard feedforward and recurrent networks.

The authors of [99] claim that an FIR neural network achieves improved prediction because the time delays within the network architecture allow it to cope with time varying signals. We have carried out several experiments applying such a neural network to the real traffic traces already discussed, however the results obtained were very similar to those using the standard feedforward neural network. Moreover, the added number of free parameters associated with an FIR network

caused it to take much longer to converge to a stable set of weights, sometimes an order magnitude longer.

4.8.2 On-Line Training

The disadvantage with off-line training is that it assumes that the traffic is stationary and that the characteristics remain constant over time. Several experiments have been carried out adopting the sliding window training approach described in Section 4.4.1.1. Over short time scales this approach yielded similar performance to off-line training, however over longer time scales it showed a slight improvement, although the neural network was still unable to pre-empt variations in the frame size. The reason for this small improvement is that over longer time scales the average frame size changes, and as was demonstrated earlier, for some traffic types the neural network tends to predict values relatively close to the mean and only reacts slowly when the frame rate changes abruptly. Therefore the on-line training scheme re-learns the mean frame size for every training set whereas the off-line method assumes the same mean throughout the entire test set.

4.9 Summary

In this chapter we have highlighted the possible advantages of having an accurate traffic prediction scheme. However, we have demonstrated that, contrary to popular belief, accurate prediction of *real* traffic using neural networks is not possible. It is however possible to achieve satisfactory prediction when considering some model generated traffic, the reason being that there is an underlying process governing the traffic generation and the neural network is able to learn this generating process. We can broadly classify these traffic generation processes into three classes : purely deterministic, purely stochastic and a hybrid.

In a purely deterministic generating process there is a strict relationship between past and future arrivals, an example of this is the chaotic time series discussed in Section 4.5.2. In this case the neural network is able to learn this relationship and thus perform very well indeed. For a purely stochastic generation process, for example the MMDP discussed in Section 4.5.3, prediction is impossible since the traffic is generated randomly. For a hybrid generating process, as in the teleconference traffic model in Section 4.5.1, the accuracy of the prediction is dependent upon the degree of randomness.

We have also demonstrated that it is possible to achieve at least as good performance with a more

simple linear regression fit compared to a neural network. The only occasion in which this was not the case was no surprise since this was the prediction of the non-linear deterministic fractal time series.

To predict real traffic the traffic arrivals must follow a deterministic process with probably an element of superimposed randomness. From the experiments carried out we conclude that such a process does not exist. However the method by which MPEG-I codes video frames gives an MPEG-I traffic stream certain characteristics which the prediction scheme is able to learn. However, it is only able to learn that the MPEG-I sequence is approximately periodic, and hence the frames sizes predicted for the next period of 12 frames are the same as sizes of the previous 12 frames. This approach is fine for video sequences of low activity, such as the video teleconference sequence discussed in Section 4.6.1, however for high activity movie sequences the prediction becomes less accurate. This characteristic of the predictions tracking the actual values is common and on no occasion was the predictor able to anticipate the traffic fluctuations. The degree with which the prediction tracked the actual sequence was dependent upon the traffic type, for some traffic types such as the aggregate group of pictures discussed in Section 4.6.2.1 the response to changes was quite slow whereas with the JPEG movie sequence in Section 4.6.2.2 the response was much faster. The reason for this was that the JPEG sequence had a tendency to have large jumps in the frame size and then remain at that rate for some time. These jumps probably indicated scene changes and frames of equal size probably represented very similar shots, thus blindly tracking the actual sequence gave the best results. By contrast, the aggregate sequence fluctuated more rapidly in a seemingly more random manner, therefore blindly tracking the actual sequence would not be as effective.

The conclusion we can draw from this chapter is that control strategies based on short term prediction of traffic are not possible and often the best indication of future traffic characteristics are current traffic characteristics. Therefore any learning based control strategies must be of the reactive type which adapt to the current network conditions. In the next chapter we introduce a scheduling mechanism that achieves this by adapting the service rate given to each flow such that the quality of service is just satisfied, thus maximising the service available to other flows.

Chapter 5

Adaptive Scheduling Using Stochastic Learning Automata

In this chapter we present a novel packet scheduler based on the use of stochastic learning automata. Initially we give some motivation followed by a brief overview of the scheme. Next we describe the scheme in more detail, discussing the various parameters associated with it and continue by demonstrating that the automaton is capable of converging to optimal probabilities for a variety of traffic models and desired performance objectives. An ability to cope with dynamic traffic environments, variable length packets and real traffic traces is then demonstrated and we continue by introducing a framed selection method which leads to improved performance. Comparisons are then made with several scheduling algorithms and based on these we propose an alternative scheduling algorithm which is capable of performing in a near optimal manner over a wide range of traffic conditions. Next we compare the scheme with the guaranteed service discipline WFQ and show that it is more suitable for supporting tolerant traffic flows. Finally we summarise the chapter and present conclusions.

5.1 Motivation

In previous chapters we have highlighted the difficulties associated with traffic characterisation and the problem this presents to the goal of achieving high network utilisation whilst providing performance guarantees. Low utilisation is inevitable when providing a deterministic service to intolerant applications. Greater utilisation is achievable by providing a statistical service, however this generally requires accurate traffic characterisation. A measurement based service is able to achieve high utilisation without requiring accurate traffic characterisation. This is at the expense of the risk of QoS violations, however some applications with soft requirements may accept this in return for a lower cost which will result from the increased utilisation. Utilisation can be maximised when each traffic flow is allocated minimal resources such that the performance objectives are just satisfied, thus maximising the available resources for other flows.

A number of measurement based admission control procedures have been proposed in the literature which work on the premise that the packets are queued according to the FIFO discipline, a more detailed discussion of these will be given in the next chapter. By contrast, few measurement based queue management schemes have been proposed. In [142] the aim is simply to satisfy mean delay requirements and under this policy an on-line measurement of the average delay is maintained for each flow. A packet belonging to the flow which has the largest ratio between the measured average delay and its delay requirement is scheduled for transmission in the next time slot. Unfortunately this scheme is limited to supporting mean delay requirements, moreover although the authors prove that it achieves optimal efficiency compared to other policies, in that it can support the highest traffic load, an admission control policy is not discussed and is not obvious, therefore the usefulness of the scheme is questionable.

In [25], a scheme is proposed to ensure that all streams experience the same loss performance and an on-line counter is used to record the number of lost packets from each stream. When a packet arrives at a full buffer, the packet belonging to the stream with the smallest counter value is discarded. This approach is only suitable when the streams have the same arrival rates and the scheme is generalised in [143] to deal with streams having different arrival rates but the same loss requirement. An on-line measurement of the loss ratio for each stream is maintained and when a packet arrives at a full buffer, the policy discards a packet that belongs to the flow with the smallest loss ratio. This approach is extended further in [144], here the streams can have different loss objectives and the when the buffer is full a packet from the stream with the smallest ratio between its loss ratio measurement and the desired loss requirement is discarded. Such approaches are limited to satisfying loss objectives and it is assumed that each flow has the same delay requirement, furthermore there is no discussion of admission control.

In the packet scheduler proposed in this chapter, a stochastic learning automaton uses measurements to adapt to the current network conditions such that performance objectives are just satisfied. The scheduler is effectively gambling that the current situation is an accurate guide to the future which is precisely the strategy suggested to support tolerant applications with soft performance requirements [4].

5.2 Overview of Novel Packet Scheduler

The packet scheduler proposed in this chapter is based upon the use of a stochastic learning automaton. An overview of the theory of automata is inappropriate since traditional notation is

based upon slightly different applications and is unsuitable in the current context, therefore we consign such a review to Appendix B.

5.2.1 Packet Scheduler

We shall assume fixed length packets and slotted time which is consistent with the ATM principle, however we shall demonstrate later that this scheme may also be applied to a system with variable packet lengths. The scheduler we are considering consists of N FIFO queues and a single server. Such a model can be considered as either an output buffer in a switch or a simple multiplexer (Figure 5.1).

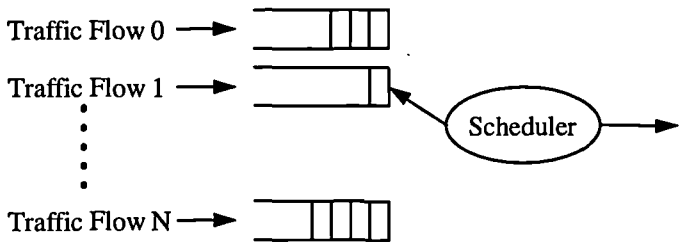

Figure 5.1:Packet scheduler model

Associated with each FIFO queue is a traffic flow and we use the term flow in a generic sense to mean either a single connection or a group of connections. Associated with flows 1 to N are particular performance objectives, however we assume that flow 0 does not have any performance objectives associated with it, this flow shall be referred to as best-effort. Although it is quite feasible for such a flow to exist, the queue associated with it is in effect an abstract concept and effectively this queue mops up any spare capacity in the system, thus we assume that there is always a packet waiting to be served in this queue. At the beginning of each time slot an arbiter must select a queue to serve and it is this decision that determines the performance observed by each flow.

5.2.2 Integration of automaton and scheduler

In the proposed scheme the actions of an automaton represent the selection of a particular queue. Later we shall present several variations on the precise scheduling algorithm, however each variation is based on the idea that a particular queue is selected with a regularity according to the probability associated with it.

The performance of each flow is measured over a measurement period, T, and the fundamental

principle behind the scheme is as follows :

If a particular flow is performing worse than requested then the service rate for that flow is increased by increasing the probability of selecting that queue and decreasing the probability of selecting the best-effort queue. Conversely, if a flow is performing better than requested then the service rate for that flow is reduced by decreasing the probability of selecting that queue and increasing the probability of selecting the best-effort queue.

Therefore the probability associated with the best-effort queue represents the spare capacity in the system and is effectively a probability pool. It is important to note that the queues with performance objectives do not increase/decrease their probabilities at the expense of other queues with performance objectives, which is generally the case with traditional automata.

The system is capable of satisfying a variety of performance objectives in terms of loss rates and delays. As with other work-conserving disciplines delay variation guarantees are loose and simply bounded by the maximum delay, however by introducing additional components such as regulators more strict delay variation requirements can be satisfied. An important fact concerning this scheme is that these performance objectives cannot be guaranteed and the scheme is only applicable to tolerant applications which can handle the occasional performance violation.

5.3 Functionality of the Scheduler

In this section we look at the workings of the scheduler in more detail and describe the two aspects that govern the behaviour of the system, the environment response and the learning algorithm.

5.3.1 Environment Response

Traditionally the environment response consists of the degree of success or failure of a particular action, with this response lying in the range [0,1] with 0 and 1 denoting the extremes of success and failure respectively. This notation of success and failure is inappropriate in the context of the proposed packet scheduling scheme, therefore a new notation is required.

Define P_i to be the requested performance index and $\hat{P}_i$ to be the measured performance index during the measurement period T for a particular flow i. The environment response will take on values in the range [-1,1] and this will reflect the difference in the actual performance and desired

performance for a particular flow. For simplicity we shall assume a linear environment response given by,

$$\beta_i = \begin{cases} 1 & \text{if } P_i - \hat{P}_i > P_i \\ \dfrac{P_i - \hat{P}_i}{P_i} & \text{if } P_i > P_i - \hat{P}_i > -P_i \\ -1 & \text{if } -P_i > P_i - \hat{P}_i \end{cases} \tag{5.1}$$

Thus a response of $\beta_i>0$ means that the performance of flow i is worse than requested, thus the probability associated with that flow must be increased. A response of $\beta_i<0$ means that the performance of flow i is better than requested, thus the probability associated with that flow can be decreased. If $\beta_i=0$ then the performance is satisfactory and no probability updates are required.

5.3.2 Learning Algorithm

The learning algorithm governs how the environment response updates the action probabilities. Depending on whether β_i is greater than or less than zero, a generic learning algorithm that can be applied to this scheme is as follows,

If $\beta_i>0$,

$$p_i(t+1) = p_i(t) + f_i(\beta_i) \tag{5.2}$$

$$p_0(t+1) = p_0(t) - f_i(\beta_i) \tag{5.3}$$

If $\beta_i<0$,

$$p_i(t+1) = p_i(t) + g_i(\beta_i) \tag{5.4}$$

$$p_0(t+1) = p_0(t) - g_i(\beta_i) \tag{5.5}$$

where p_0 is the probability of serving the best-effort queue and the functions f_i and g_i are the reward and penalty functions respectively. The terms reward and penalty are a slight misnomer in this context since both are applied when the system is performing in a sub-optimal manner, therefore they should simply be considered as the functions that result in probability increases and probability decreases respectively.

Traditionally the functions f_i and g_i are given by,

$$f_i(\beta_i) = a(1 - p_i(t))\beta_i \tag{5.6}$$

$$g_i(\beta_i) = bp_i(t)\beta_i \tag{5.7}$$

where a and b are the reward and penalty parameters respectively. The restriction that a and b are in the range [0,1] ensures that probabilities associated with each action are always in the range (0,1) and that they approach these values asymptotically.

5.3.3 System Parameters

The parameters that govern system behaviour are the measurement interval, T, the reward parameter, a, and the penalty parameter, b. These parameters govern the convergence behaviour of the automaton and there are two extremes of behaviour regarding convergence. The first extreme is when the convergence is slow, however when the optimal probability is reached the probability remains relatively stable. The other extreme is when convergence to the optimal is fast, however the probability subsequently fluctuates about this optimal. These two extremes are illustrated in Figure 5.2 where the initial probability is 1 and the optimal probability is 0.5.

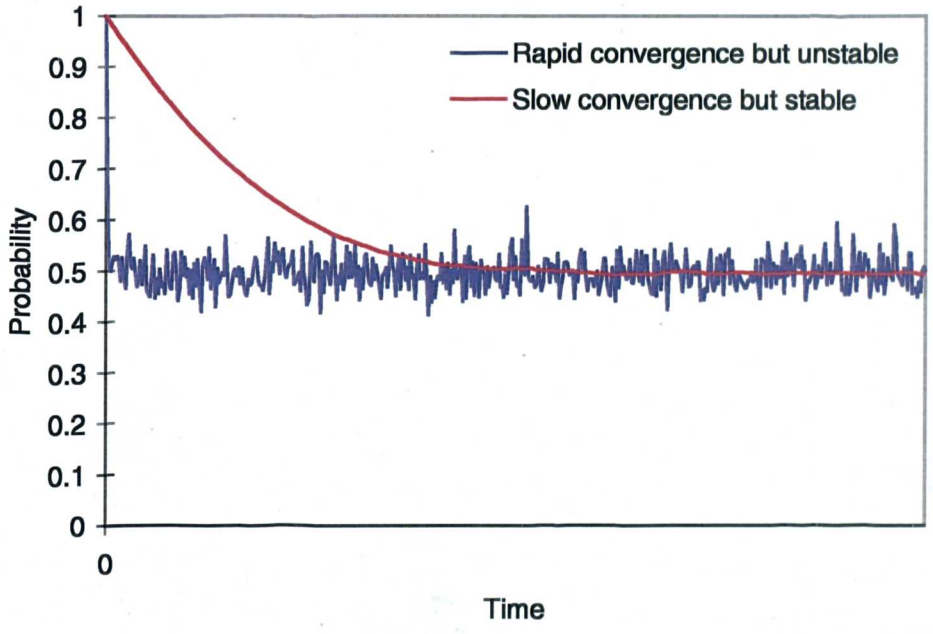

Figure 5.2:Extreme convergence behaviour of learning automata

We can relate this general behaviour to the proposed scheme, for example a short measurement interval, T, results in regular probability updates which consequently leads to the probabilities quickly approaching the optimal values. However, the stochastic nature of arrivals and departures implies that there will be some degree of variance in the measured performance over

this interval. By the law of large numbers, the shorter the measurement interval the greater the observed variance, therefore the probabilities will change on a regular basis and oscillations about the optimal will occur. By contrast, a longer measurement interval will reduce this variance leading to more stable probabilities at the cost of taking longer to converge. The optimal measurement interval is highly dependent upon the nature of the traffic and must be sufficiently long to have confidence in the measurement yet short enough to ensure rapid convergence. One may argue that it is more suitable to update the probability associated with a particular flow after a given number of packets belonging to that flow have been served, therefore allowing some degree of confidence in the measurement to be established. However, such an approach would introduce a degree of unfairness in that flows with high service rates would receive more regular updates than those with lower service rates. If for instance the multiplexer became overloaded then it would be the flows with the higher service rates that would quickly consume any spare capacity.

The parameters a and b determine the actual change in probability given the difference in measured performance to the desired performance. As with small values of the measurement interval T, large values of a and b result in the automaton quickly approaching the optimal followed by oscillations about the optimal. The converse is true for small values of a and b. More precisely, the overall convergence properties of the automaton are determined by the ratios T/a and T/b.

5.3.4 Initial Conditions

Since the best-effort traffic is the least important we initially set the probability associated with this queue equal to zero i.e. $p_0(0)=0$. The remaining probability is distributed equally between the other flows, therefore given that there are N traffic flows the initial action probability vector is given by,

$$\mathbf{p}(t=0)=\left(0,\frac{1}{N-1},\ldots\ldots,\frac{1}{N-1}\right) \tag{5.8}$$

This choice of initial probabilities is purely arbitrary and this issue is discussed more thoroughly in the context of admission control in Chapter 6.

5.4 Convergence to Theoretical Optimum

In this section we present results for various traffic scenarios in which it is possible to analyse the behaviour of the system analytically and hence given the performance objectives it is possible to calculate the optimal probability vector beforehand. In order to create an analytically tractable scenario we need to assume the following scheduling algorithm :

The automaton stochastically selects a particular queue for service. If this queue is occupied then the packet at the front of the queue is served, otherwise the packet at the front of the best effort queue is served.

In the rest of this thesis we shall refer to this algorithm as the 'original scheduling algorithm' and it allows us to model each queue as a separate queue having a discrete time server of rate $\mu_i=p_i$, where p_i is the probability associated with the particular queue i.

5.4.1 Bernoulli Arrival Process

The average arrival rate at queue i is given by λ_i and this represents the probability that a packet arrives in any particular time slot. The analysis of this queueing system is presented in Appendix D.

5.4.1.1 Mean Delay Requirements

Initially we shall assume that the performance objectives are in terms of a mean delay, and we denote D_i and $\hat{D}_i$ to represent the requested delay and measured delay in time slots respectively. The traffic environment we shall consider is arbitrary and consists of three traffic sources, two of which have mean delay requirements and the other is best effort. The traffic environment is summarised in Table 5.1.

Traffic Flow, i	λ_i	D_i	μ_i
0	-	-	-
1	0.65	6	0.7
2	0.1	8	0.2

Table 5.1:Summary of traffic environment for initial experimentation

The system parameters are T=200 and a=b=0.005 and Figure 5.3 shows the probability

associated with each queue and the calculated optimal plotted against time. The resulting performance after the initial convergence is summarised in Table 5.2.

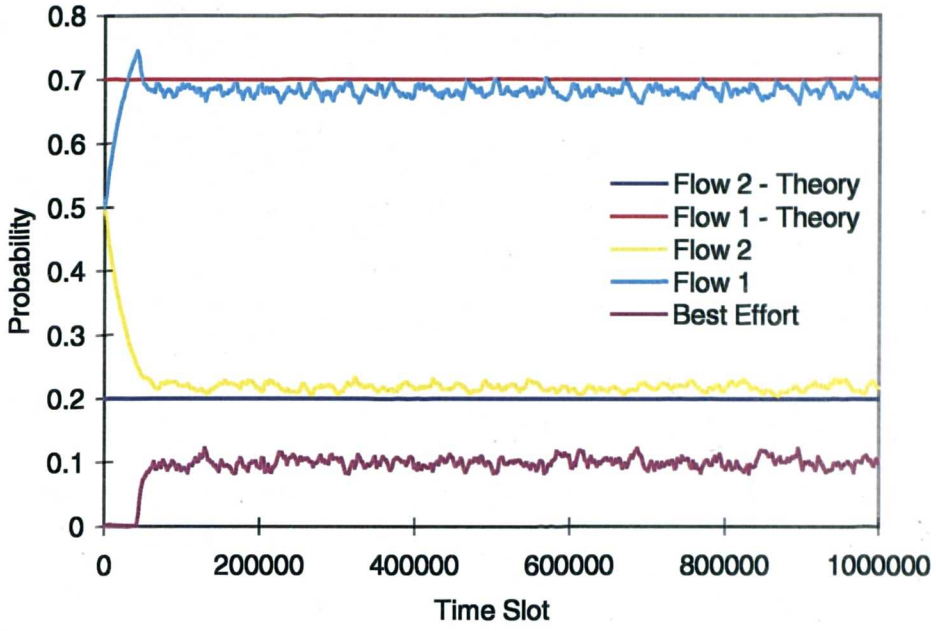

Figure 5.3:Probability plot using original learning algorithm for traffic given in Table 5.1

Traffic Flow, i	D_i	$\hat{D}_i$ (95% C.I)
1	6	10.20±0.29
2	8	6.43±0.35

Table 5.2:Performance of original learning algorithm for traffic given in Table 5.1

Clearly the performance objectives of flow 2 are comfortably satisfied whereas those for flow 1 are not, this characteristic is a consequence of the choice of learning algorithm. According to this learning algorithm the probabilities are updated such that they approach 0 and 1 asymptotically. Usually this is a desirable feature of learning automata since it prevents them from becoming locked in a state with a probability of 0 or 1. A result of this is that the nearer the probability is to 0 (1) then the greater the average probability increase (decrease) and the smaller the average probability decrease (increase). Evidence of this is apparent in Figure 5.3 where close inspection shows that the probability associated with flow 1 is generally less than the calculated optimal and the probability associated with flow 2 is generally more than the calculated optimal. Therefore we choose to remove this asymptotic characteristic by modifying the learning algorithm such that the probability updates are not dependent upon the current probabilities. Furthermore we relax the requirement that the environment response must lie in the range [-1,1] since this allows

greater probability changes should the measured performance be far from the desired performance. The environment response is now given by,

$$\beta_i = \frac{P_i - \hat{P}_i}{P_i} \tag{5.9}$$

and the update algorithm becomes,

If $\beta_i > 0$,

$$p_i(t+1) = Min\{p_i(t) + a\beta_i, 1-(N-2)\Delta\} \tag{5.10}$$

$$p_0(t+1) = Max\{p_0(t) - a\beta_i, 0\} \tag{5.11}$$

If $\beta_i < 0$,

$$p_i(t+1) = Max\{p_i(t) + b\beta_i, \Delta\} \tag{5.12}$$

$$p_0(t+1) = Min\{p_0(t) - b\beta_i, 1-(N-1)\Delta\} \tag{5.13}$$

Since the probabilities approach 0 and 1 directly and not asymptotically then the *Min* and *Max* functions are required to ensure that $\Delta \leq p_i(t) \leq 1$. The term Δ is the minimum probability that can be associated with a particular queue, this term is required to ensure that the probability does not fall to zero thus leading to the queue never being selected for service. A plot of probability against time using this learning algorithm with the traffic given in Table 5.1 is presented in Figure 5.4. In this situation the reward and penalty parameters have both been reduced to 0.0025 in order to ensure that the probability updates remain in the same range as before.

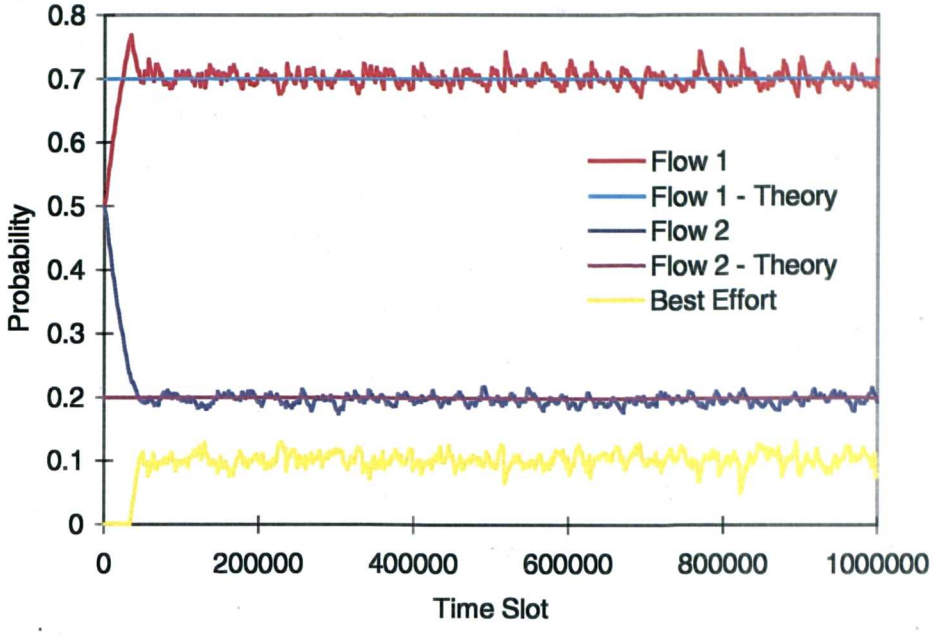

Figure 5.4:Probability plot using modified learning algorithm for traffic given in Table 5.1

Traffic Flow, i	D_i	$\hat{D}_i$
1	6	6.11±0.03
2	8	8.30±0.05

Table 5.3:Performance of modified learning algorithm for traffic given in Table 5.1

This learning algorithm does not suffer from the obvious bias of the original learning algorithm. However from the resulting performance presented in Table 5.3 it is clear that none of the performance objectives are met. The reason for this is the geometric delay distribution of each flow. The delay distribution of any flow is geometric whenever the service follows a Bernoulli process, regardless of the arrival process [145]. This geometric delay distribution gives rise to a relationship between the mean delay and service rate given in Figure 5.5 (Appendix D). Clearly a reduction in the service rate has a more marked effect on the delay than an equivalent increase in service rate. Therefore if the service rate oscillates evenly about the optimal then the periods when the service rate is less than the optimal will influence the overall mean delay more than the converse, thus resulting in a mean delay greater than desired.

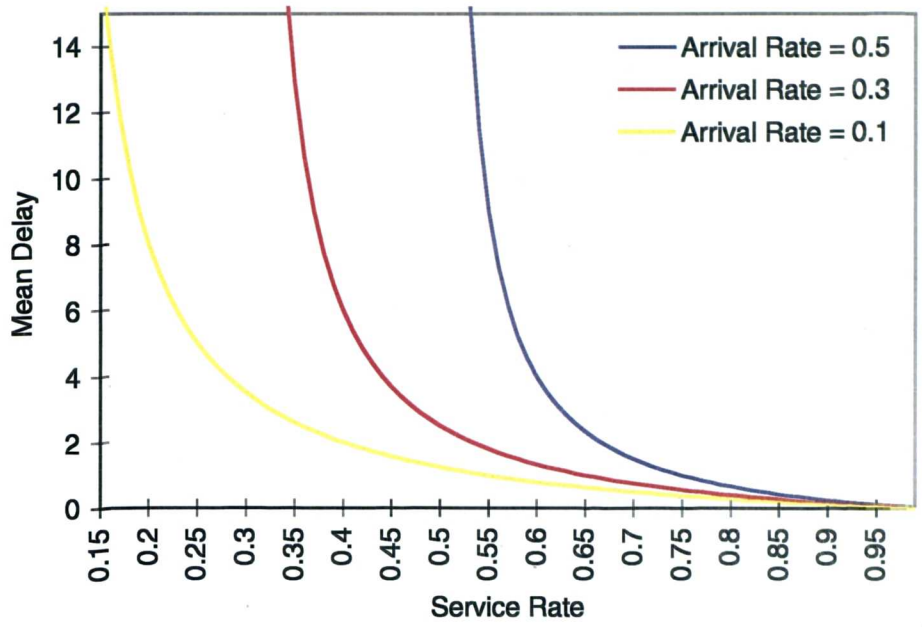

Figure 5.5:Plot of mean delay versus service rate for Bernoulli arrivals of various rates

To overcome this shortcoming it is necessary to increase the reward parameter with respect to the penalty parameter such that in the long run the probability increases are greater than the probability decreases, thus resulting in satisfactory performance. Table 5.4 shows the resulting performance for various parameter pairs for the traffic scenario described in Table 5.1.

a	b	D_1	$\hat{D}_1$	D_2	$\hat{D}_2$
0.0025	0.0025	6	6.11±0.03	8	8.30±0.05
0.0025	0.0020	6	5.71±0.04	8	7.93±0.04
0.0025	0.0015	6	5.25±0.03	8	7.49±0.04
0.0025	0.001	6	4.70±0.04	8	6.93±0.05
0.0025	0.0005	6	3.95±0.05	8	6.07±0.06

Table 5.4:Performance of modified learning algorithm for traffic given in Table 5.1 with various penalty parameters

Clearly it is the ratio between the reward and penalty parameters that govern if and by how much the performance objectives are met. In this case parameters of 0.0025 and 0.002 result in satisfactory performance, however the required ratio is likely to depend on factors such as the arrival characteristics and performance objectives. A large conservative choice for this ratio will often ensure that the performance objectives are satisfied, however this decision must be made by

the network operator and determines how much the network is prepared to gamble.

5.4.1.2 Maximum Delay Requirements

In the previous section we demonstrated the performance of the learning automaton packet scheduler when the objectives were in terms of a mean delay. Although such performance objectives are likely to play a part in future integrated services networks, a more appropriate performance objective with respect to the applications the scheduler is aimed at is in terms of a maximum delay. As mentioned previously, applications that require strict bounds on the maximum delay will declare their traffic parameters in term of a deterministically bounded traffic model and will be scheduled using service disciplines such as WFQ. However, tolerant applications will not require such strict bounds and their performance objectives will be statistical in nature, for example a given fraction of packets must meet a maximum delay bound.

These performance objectives are catered for by simply measuring the fraction of packets that miss this deadline and updating the probabilities based on this measurement. A similar approach can be applied when the performance objective is in terms of a buffer overflow rate, since in this case the number of packets discarded is used as the basis of the probability updates. We introduce the new notation of L_i and $\hat{L}_i$ to denote the tolerated and measured fraction of late/lost packets. Given the traffic scenario summarised in Table 5.5 the probability associated with each flow is plotted against time with control parameters given by T=1000, a=0.0025 and b=0.0015. As mentioned previously, such parameters are not considered optimal since the definition of optimality is unclear, however they are satisfactory in the sense that they result in the performance objectives being satisfied once convergence has occurred (Table 5.6).

Traffic Flow, i	λ_i	D_i	L_i	μ_i
0	-	-	-	-
1	0.2	12	0.01	0.439
2	0.2	10	0.1	0.351

Table 5.5:Summary of traffic environment used to satisfy maximum delay objectives

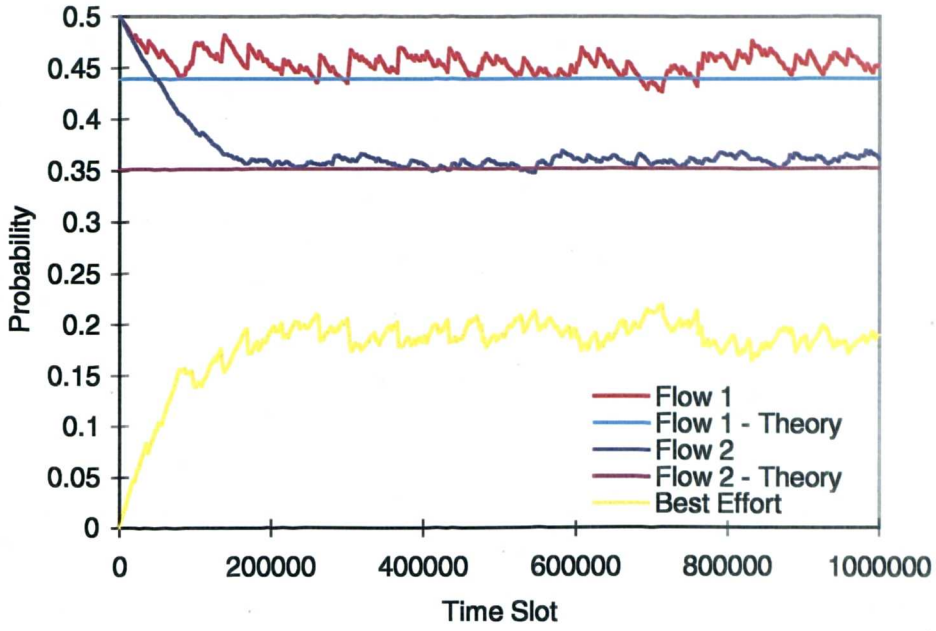

Figure 5.6:Probability plot for traffic given in Table 5.5

Traffic Flow, i	D_i	L_i	$\hat{L}_i$
1	12	0.01	0.0076
2	10	0.1	0.0981

Table 5.6:Performance for traffic given in Table 5.5

Although so far we have assumed that the flows all have a mean delay requirement or they all have a maximum delay requirement, the scheduler is able to support a heterogeneous mix of these objectives since the probability associated with a particular flow is updated independently of the others.

5.4.2 Markov Modulated Deterministic Process

In the previous section the traffic environment is very simple in which there is no correlation between arrivals. It is expected that traffic carried on integrated service packet networks will however be correlated and bursty [21]. Therefore in this section we apply the scheduler to traffic which is governed by a MMDP. A MMDP is a traffic model in which the generating process alternates between two states, the OFF state corresponds to no packet arrivals and the ON state corresponds to a continuous stream of arrivals at the link rate. The periods of time in the ON and OFF states are geometrically distributed and the two parameters that describe the arrival process are the average arrival rate (λ_i) and the average ON period ($1/\alpha_i$). Further theory and queueing

results are derived in Appendix D.

Given the traffic environment in Table 5.7, the resulting probability plot and performance after the initial convergence are given in Figure 5.7 and Table 5.8 respectively. Clearly the automaton is able to converge to the optimal probabilities and provide satisfactory performance.

Traffic Flow, i	λ_i	$1/\alpha_i$	D_i	L_i	μ_i
0	-	-	-	-	-
1	0.1	4	15	0.1	0.422
2	0.15	2	18	0.1	0.303

Table 5.7:Summary of traffic environment for MMDP experiments

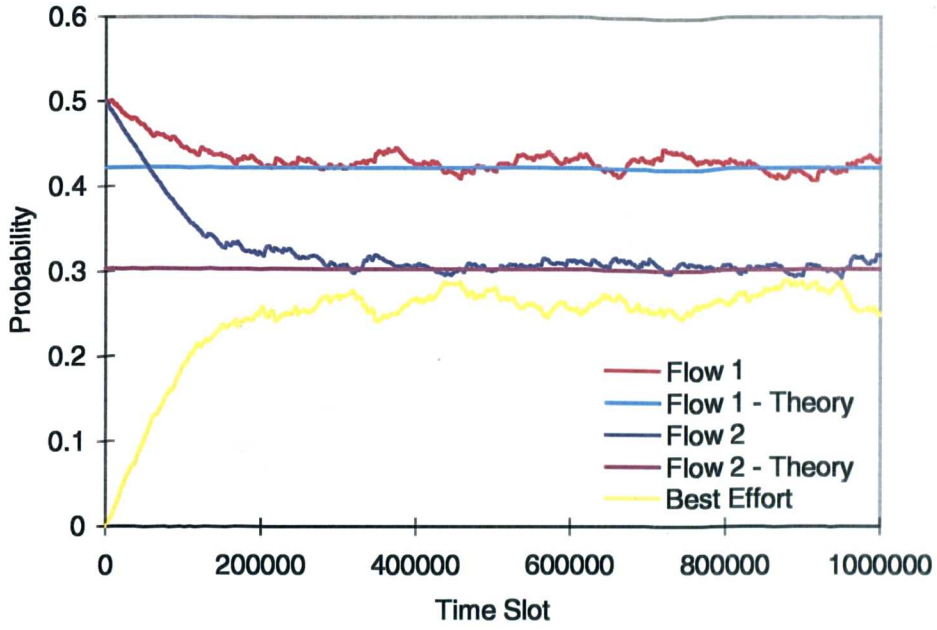

Figure 5.7:Probability plot for traffic given in Table 5.7

Traffic Flow, i	D_i	L_i	$\hat{L}_i$
1	15	0.1	0.099
2	18	0.1	0.098

Table 5.8:Performance for traffic given in Table 5.7

5.4.3 Dynamic Traffic Environment

One of the main advantages of the learning automaton scheme is that it is able to adapt should the

traffic environment change. For example the arrival rate of a particular flow may change and/or the performance objective associated with it may also change. The arrival rate of a flow may vary because a reactive congestion control scheme detects the onset/termination of a congestion period and requests an increase/decrease in transmission rate. Alternatively the transmission rate of a video sequence may change because of scene changes consisting of different activity. The performance objectives may differ because a tolerant playback application could alter its playback point such that performance is not permanently degraded.

In order to model a dynamic traffic environment we assume the two traffic sources switch characteristics midway through the simulation. Therefore the arrival rate and mean delay objective of flow 1 becomes the arrival rate and mean delay objective of flow 2 and vice versa. The probability plot is shown in Figure 5.8 and clearly the automaton is able to adapt to the change.

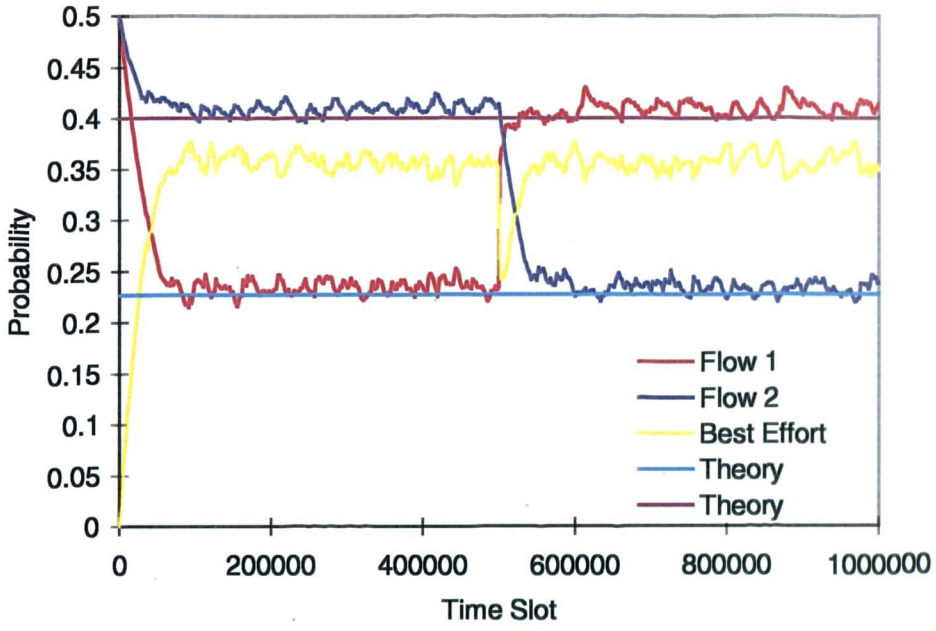

Figure 5.8:Probability plot in a dynamic traffic environment

Although throughout we have assumed that the best effort queue is permanently occupied, equivalent performance may be observed if we relax this assumption providing that if an empty best effort queue is selected, the server remains idle for that time slot. One may argue that making the server non-work conserving may degrade the performance of the other flows. This is not the case since the other flows will already be receiving sufficient service and should their performance degrade, the automaton will simply adapt by allocating more probability to these flows at the expense of the best effort flow.

5.5 Application to Real Traffic

In this section we apply the scheduler to a situation in which the arriving traffic stream is an actual video sequence. The sequence is an approximately 40 second long section from the JPEG encoded Star Wars trace described in Chapter 4, the frame sizes are plotted in Figure 5.9. Obviously in this scenario mathematical analysis is not possible, however our claim that the automaton continuously converges to the optimal probability is justified by the delay performance obtained.

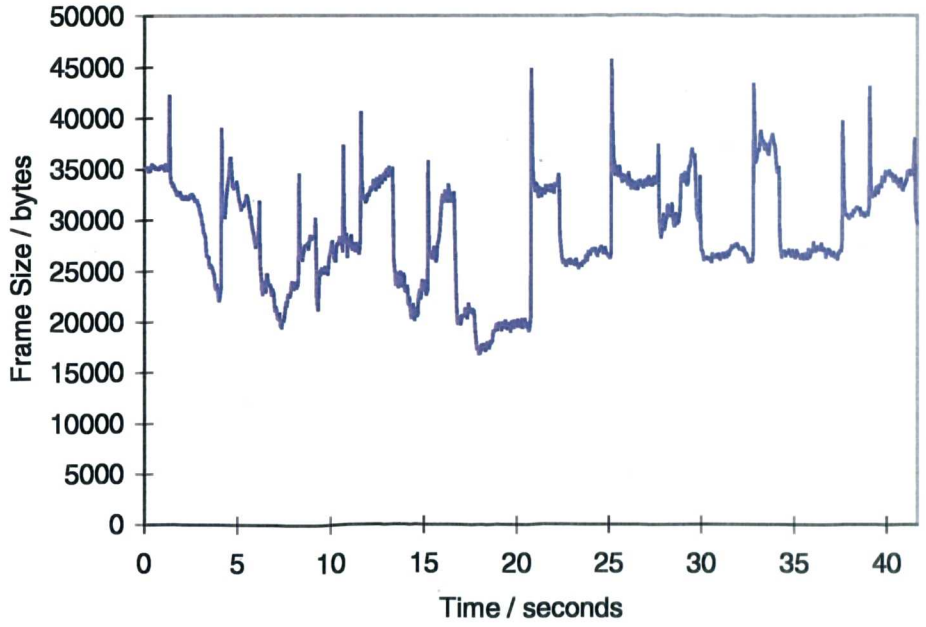

Figure 5.9:JPEG encoded section from Star Wars

A frame is generated every 1/24th of a second (41ms) and we assume that when a frame is generated it is split into ATM cells of 48 bytes and these cells are transmitted evenly over the 1/24th second interval. The maximum frame size during the whole 2 hour trace is 78459 bytes and this corresponds to a peak rate of 15Mbit/s, in order to accommodate this we set the capacity of the multiplexer to be 30Mbit/s and assume only the Star Wars and best effort traffic flows are present. The performance objective is that 0.9 of the cells suffer a delay of 500μs or less, which corresponds to about 39 cell slots. The parameters are T=8.33ms, a=0.0025, b=0.0015 and Figure 5.10 shows the probability associated with the Star Wars sequence, the corresponding plot for the best effort flow is omitted for clarity. After the initial convergence which takes approximately 5 seconds, this parameter choice results in 0.926 of the cells meeting the delay requirement. Once again we do not claim that this parameter set is optimal and various other choices would result the requirements being satisfied, again the decision as to the most

appropriate parameter set is left to the network operator.

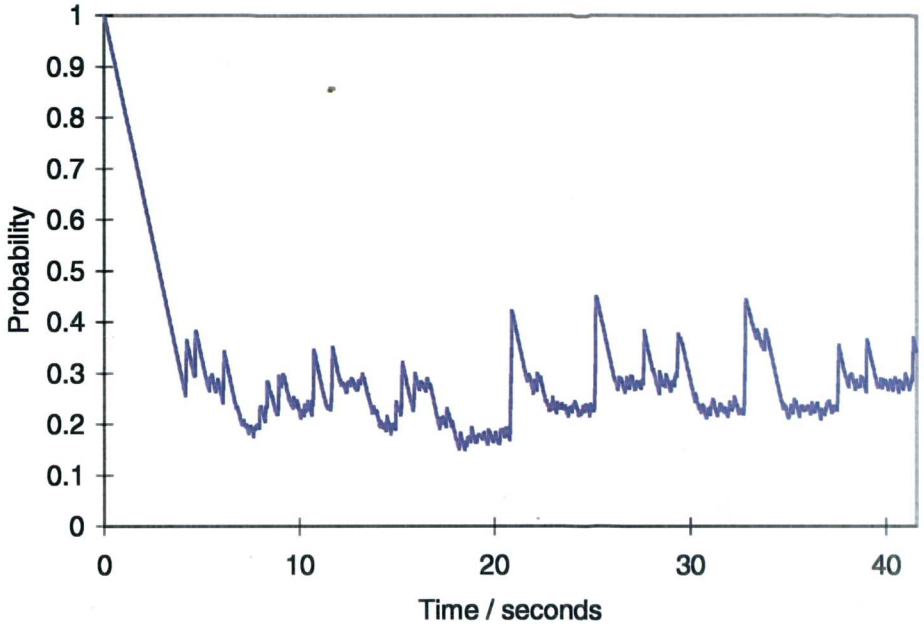

Figure 5.10:Probability plot for Star Wars

5.6 Application to Variable Length Packets

So far we have assumed that each packet is of constant length and takes the same amount of time to be serviced, such an assumption is consistent with the ATM principle. The IntServ principle however is based upon variable length packets, the service time of which is proportional to the length of the packet. In this section we assume that the length of each packet, in bytes, follows a geometric distribution of mean S_i. In order to be consistent with previous experiments the multiplexer capacity is such that it may service 53 bytes in a single time slot. The traffic scenario is summarised in Table 5.9 and the corresponding probability plot with parameters T=1000, a=0.0025 and b=0.0015 is given in Figure 5.11. After the initial convergence we observe satisfactory mean delays of 7.47 and 4.73 for flows 1 and 2 respectively.

Traffic Flow, i	λ_i	D_i	S_i
0	-	-	-
1	0.2	8	106
2	0.2	5	53

Table 5.9:Summary of traffic in variable packet length experiment

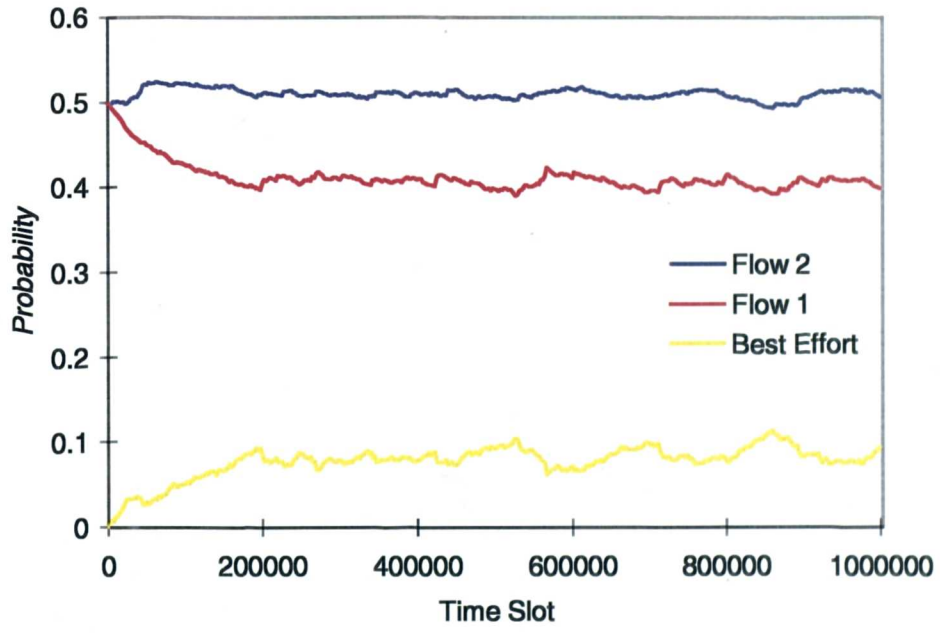

Figure 5.11:Probability plot for variable packet length experiment

5.7 Framing Mechanism

In the scheme described so far the queues are selected stochastically according to the probability vector. Now according to the laws of probability, it is feasible that a queue with a rather large probability may be starved for a significant amount of time or a queue with a small probability may be selected in two consecutive time slots. In order to overcome this variability in the selection procedure we introduce the notion of frames which are based on the probability vector. A frame is essentially a list of queue indices and the proportion of elements corresponding to a particular queue is approximately equal to the probability associated with that queue. At the start of every time slot the queue corresponding to the next element in the frame is selected as opposed to selecting the queue stochastically.

5.7.1 Frame Generation Algorithm

The length of a frame is F time slots and at each frame generation instant flow i is allocated R_i time slots in the frame and unlike simple round robin scheduling in which R_i has to be an integer, we allow R_i to be any real number. This modification allows the granularity of bandwidth allocation to be arbitrarily small, irrespective of the frame length. The goal of the framing mechanism is to allocate flow i the integer part of R_i slots in each frame and exactly R_i slots per frame in the long run. In order to achieve this we need to associate a variable r_i with flow i which represents the fractional part of R_i. Therefore at the nth frame generation instant, the variables are updated as follows,

$$R_i(n) = F.p_i(n) + r_i(n-1) \tag{5.14}$$

$$r_i(n) = R_i(n) - \lfloor R_i(n) \rfloor \tag{5.15}$$

resulting in $\lfloor R_i \rfloor$ time slots being allocated to flow i in this frame. To ensure that queues are selected uniformly during the frame, the time slots are allocated in increasing order of $m_i / \lfloor R_i \rfloor$, where m_i is the number of time slots already allocated to flow i. The interval between frame generation instants can be arbitrary providing that they are constant throughout, although initially we shall assume a new frame is generated after a single pass though the current frame. After every probability update the r_i's are set to zero and the process starts over.

Consider a frame length F=10 and three active flows with p_0=0.1, p_1=0.55 and p_2=0.35. This probability vector allocates $\lfloor R_0 \rfloor = 1, \lfloor R_1 \rfloor = 5$ and $\lfloor R_2 \rfloor = 3$ time slots to each flow resulting in

r_0=0, r_1=0.5 and r_2=0.5. Therefore the first frame is,

0	1	2	1	2	1	1	2	1

At the next frame generation instant the variables are updated such that R_0=1, R_1=6 and R_2=4 which leads to integral $\lfloor R_i \rfloor$'s and a second frame of,

0	1	2	1	2	1	1	2	1	2	1

5.7.2 Performance of framing mechanism

Applying this scheme with F=30, T=200, a=0.0025 and b=0.0015 to the traffic scenario described in Table 5.1 we find that the probabilities required such that the performance objectives are satisfied are lower than the theoretical probabilities calculated when the original stochastic selection procedure is adopted (Figure 5.12). This results in an increased probability associated with the best-effort queue and means there is greater spare capacity in the system. A further advantage of this scheme is that there is no need to generate random numbers, thus reducing the computational burden. One may argue that the rather complicated frame generation process counteracts this, however frame generation may be performed off-line.

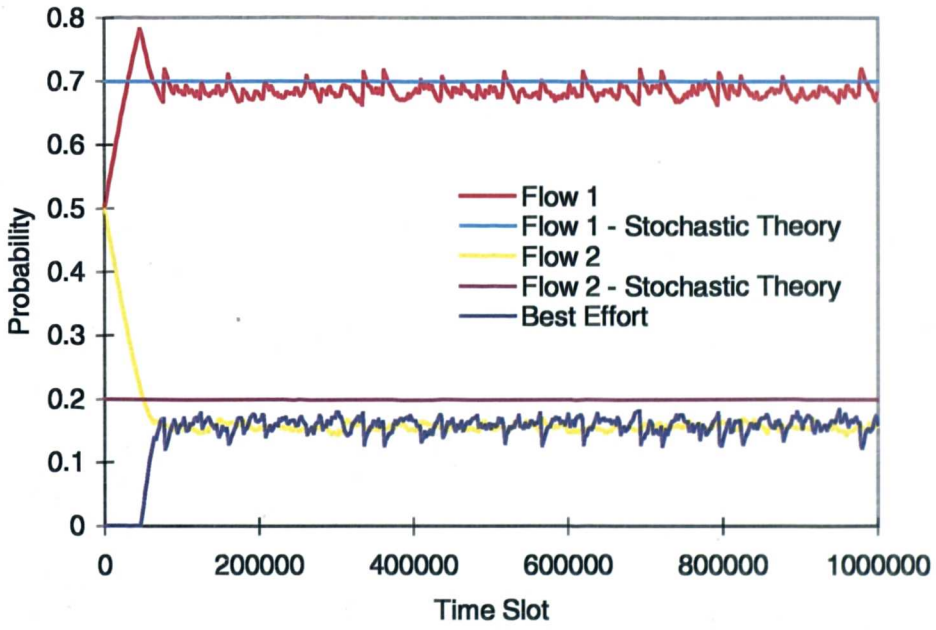

Figure 5.12: Probability using framed selection method

5.8 Comparison with Alternative Algorithms

We have demonstrated that the proposed learning automaton scheduling mechanism is capable of converging to the optimal probabilities. This results in satisfactory performance and maximises the spare capacity in the system, which in turn may lead to higher utilisation. Various methods of comparing scheduling algorithms exist [40], however from a purely performance perspective they should be compared by the amount of traffic they can withstand whilst satisfying the performance objectives of the flows they support. Initially we shall highlight a shortcoming of the original scheduling algorithm described in Section 5.2.2, then present various alternatives that overcome this and compare these with existing scheduling algorithms.

5.8.1 Shortcomings of Original Scheduling Algorithm

Adopting the traffic scenario described in Table 5.10 we demonstrate the shortcomings of the original algorithm by comparing its performance with alternative algorithms when the arrival rate of flow 1 is fixed and that of flow 2 is varied.

We compare the automaton scheme with Static Priority (SP), Earliest Deadline First (EDF) and First-In-First-Out (FIFO). With SP priority is always given to the flow with the more stringent delay requirement and with EDF we interpret the deadline to be the mean delay requirement. Figure 5.13 shows a plot of the mean delay against flow 2 arrival rate whilst keeping the arrival rate of flow 1 fixed at 0.3.

Traffic Flow, i	λ_i	D_i
0	-	-
1	0.3	4
2	-	3

Table 5.10:Summary of traffic scenario used for comparison

Clearly the automaton scheme is unable to withstand the same load as the other schemes and fails to meet the performance objectives at arrival rates of approximately 0.41 and 0.51 for the stochastic and framed selection methods respectively. By contrast EDF is able to withstand an arrival rate of approximately 0.64. Although EDF yields the best performance, followed by FIFO and SP, it is incorrect to conclude that this is always the case and different traffic scenarios lead to different results. In this situation, since the objectives of the two streams are similar the

FIFO algorithm outperforms SP, however we shall see later that when the performance objectives differ then SP is superior.

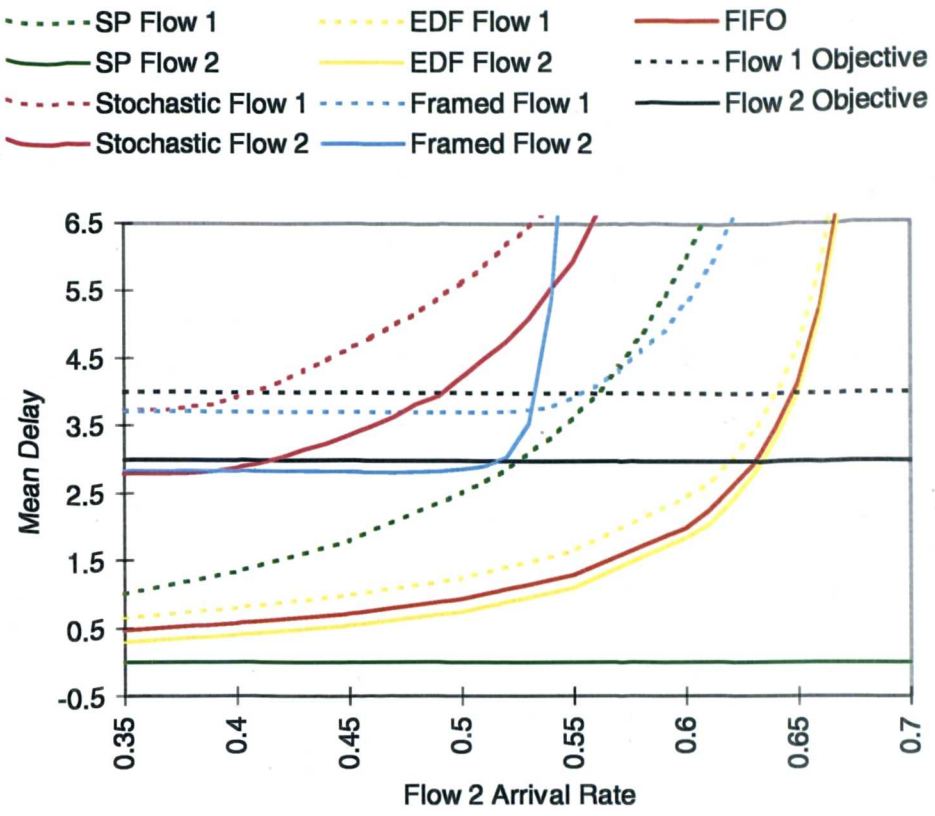

Figure 5.13:Comparison of performance of automaton scheme with alternative algorithms

The reason for the shortcoming of the automaton scheme is due to the scheduling algorithm adopted. In this scheduling algorithm, if the selected queue is empty queue the best effort queue is served. The stochastic nature of the arrivals ensures that a queue chosen for service will often be empty resulting in the best effort flow being served, even though other queues may be occupied. This characteristic leads to a situation in which the best effort flow receives more service than the probability associated with it would suggest and conversely the other queues receive less service. Therefore under congested conditions the best effort flow still receives some service when ideally it should be receiving minimal service such that greater capacity may be allocated to the other flows. This leads to performance violations at loads lower than with rival scheduling algorithms. In order to overcome this we require a scheduling algorithm that serves the best-effort queue in proportion to the probability associated with it. In addition to increasing the sustainable load, this characteristic is essential for admission control which will be discussed in the next chapter.

We now propose six alternatives differing in complexity and accuracy, where an accurate scheme is defined to be one which serves the best-effort queue exactly as suggested by the probability.

Iterative Stochastic Selection (ISS)

If an empty queue is selected then another queue is selected stochastically until a non-empty queue is found. This scheme is highly accurate however the continuous generation of random numbers may introduce a considerable processing overhead.

Iterative Framed Selection (IFS)

This is the framed alternative to ISS in which queues are selected according to the current frame until a non-empty queue is found. As with ISS, this scheme is highly accurate yet it is less complex due to the absence of any random number generation, however it still may require a number of queues to be selected before a non-empty one is found. This potential overhead could be eliminated in both ISS and ISF by introducing a limit on the number of non-empty queues that can be selected, once this limit is reached the best effort queue is selected. This limit however reduces the accuracy of the two schemes.

Stochastic Selection with Round Robin (SSRR)

If an empty queue is selected then the scheduler defaults into a round robin selection procedure, polling queues in order until a non-empty queue is found. This scheme is relatively simple, however the round robin nature allows the best effort queue to receive service even though it has not been selected directly.

Framed Selection with Round Robin (FSRR)

This is the framed alternative to SSRR and it inherits the advantages and disadvantages of this scheme.

Stochastic Selection with Round Robin excluding Best Effort (SSRRBE)

This is similar in nature to SSRR however the best effort queue is excluded from the round if an empty queue is selected initially. This scheme achieves much greater accuracy than SSRR with little added overhead.

Framed Selection with Round Robin excluding Best Effort (SSRRBE)

This is the framed alternative to SSRR and as with the other framed schemes it exhibits similar properties to its stochastic counterpart.

Under the traffic conditions previously described in Table 5.10 we compare the sustainable load of the framed (Figure 5.15) and stochastic (Figure 5.14) alternatives.

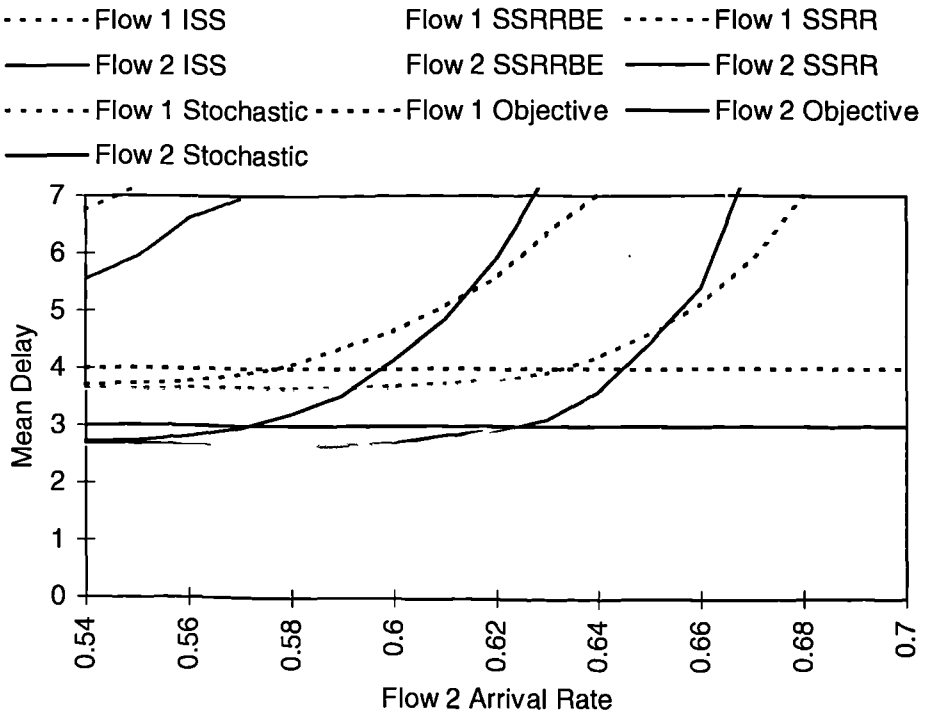

Figure 5.14:Comparison of stochastic based algorithms

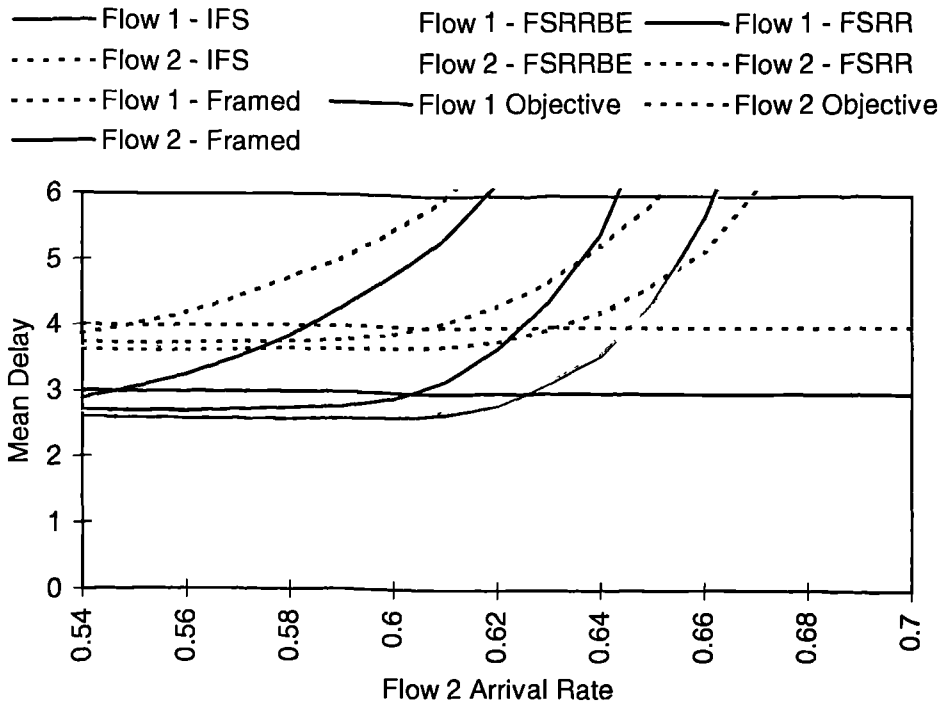

Figure 5.15:Comparison of frame based algorithms

In both cases we find that the best performance is achieved with the iterative selection processes, ISS and IFS, however very similar performance is achieved with the less complex SSRRBE and FSRRBE algorithms. The pure round robin algorithms of SSRR and FSRR give the next best performance while the original stochastic and framed algorithms result in the worst performance. Comparing corresponding stochastic and frame based algorithms we find that the framed based variation is superior for the original and round robin algorithms, however for the round robin without best effort and iterative schemes the difference is minor. Based on these results we conclude that the algorithms based on round robin without best effort are the best in terms of performance and complexity, however the difference between the stochastic and framed variation of this algorithm is small.

5.8.2 Performance Evaluation of SSRRBE

In this section we compare the performance of SSRRBE with SP, EDF and FIFO in three different traffic scenarios, each of which has been tailored such that either SP, EDF or FIFO perform very well. For simplicity we continue to focus on the scenario in which there are only two flows and we assume that the performance objectives are in terms of a maximum delay.

5.8.2.1 Traffic Scenario 1

The conditions under which SP significantly outperforms EDF and FIFO are when the performance objectives of the two flows are very different. In this case the SP scheme gives higher priority to the flow with the most stringent objectives, which in this traffic scenario is flow 2. The traffic scenario is summarised in Table 5.11. Holding the arrival rate of flow 1 at 0.3 the arrival rate of flow 2 increased and the maximum sustainable load in each case is shown in Figure 5.16.

Traffic Flow, i	λ_i	D_i	L_i
0	-	-	-
1	0.3	15	0.15
2	-	1	0.01

Table 5.11:Traffic scenario 1

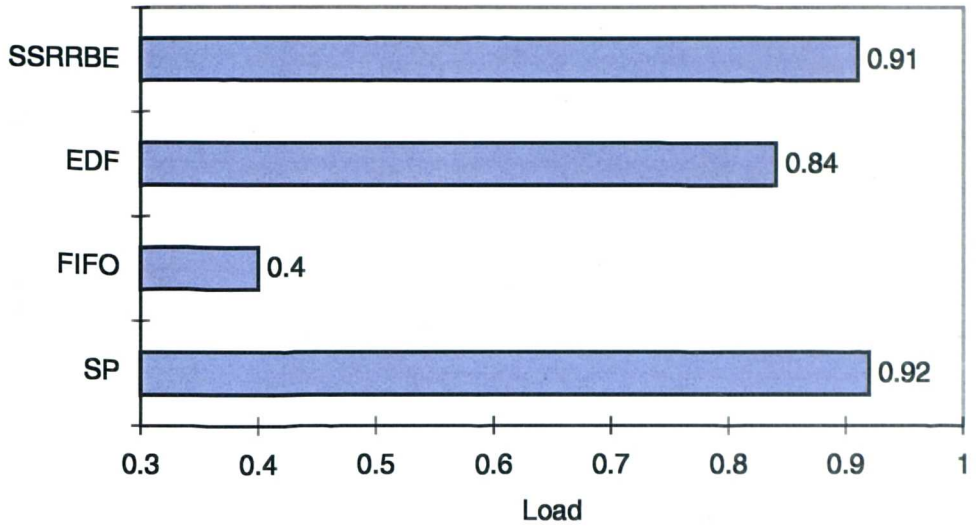

Figure 5.16:Maximum sustainable load under traffic scenario 1

As anticipated, SP significantly outperforms both EDF and FIFO, the reason being that SP always gives maximum service to flow 2 and allocates any spare capacity to flow 1. Since flow 2 is very demanding in terms of service, such treatment is necessary and the left over capacity is sufficient to support the less demanding flow 1, even at higher loads. By contrast FIFO does not differentiate between the flows, hence all packets must meet the most stringent objective in order to meet the requirements. Therefore FIFO is judged to have failed when 0.01 of all packets are delayed more than 1 time slot, despite the fact that the requirements of flow 1 are still comfortably satisfied. EDF performs better than FIFO, however EDF acts to minimise the overall loss rate of the packets and increasing the load has the same effect on the loss rate of both flows. Therefore EDF is judged to have failed as soon as the overall loss rate is above 0.01, despite the satisfactory performance of flow 1. SSRRBE performs almost as well as SP since as the load increases, the automaton simply adjusts the probabilities such that the performance objectives of both flows are satisfied. Therefore each flow is treated equally compared to its requests and this results in the situation where at one particular load, neither flow is able to meet its performance objectives. SSRRBE does not perform as well as SP because as the maximum sustainable load is approached, the best effort probability should fall to zero. However, the variance in the measured performance leads to the best effort probability occasionally becoming greater than zero, thus resulting in the occasional undesirable service of the best effort queue.

5.8.2.2 Traffic Scenario 2

The EDF algorithm acts to minimise packet loss over all flows, therefore this algorithm will perform in an optimal manner when the desired packet loss for each flow is equivalent, regardless of the maximum queueing delay. Such a traffic environment is summarised in Table 5.12 and the corresponding maximum sustainable loads are given in Figure 5.17.

Traffic Flow, i	λ_i	D_i	L_i
0	-	-	-
1	0.3	12	0.1
2	-	4	0.1

Table 5.12:Traffic scenario 2

SP does not perform as well in this case because it provides maximum service to flow 2 and the spare capacity to flow 1. This high level of service is unnecessary and leads to the starvation of flow 1 at higher loads, thus resulting in a failure to meet the performance objectives of flow 1. In a similar way to traffic scenario 1, all packets must meet the most stringent performance objectives for a FIFO scheduler to be regarded as successful and this results in a failure to meet flow 2 objectives at higher loads, despite comfortably satisfying flow 1 objectives. SSRRBE is able to adapt such that the performance objectives are satisfied resulting in a greater sustainable load, however the fluctuations in the best-effort probability result in performance just below that of the optimal EDF algorithm.

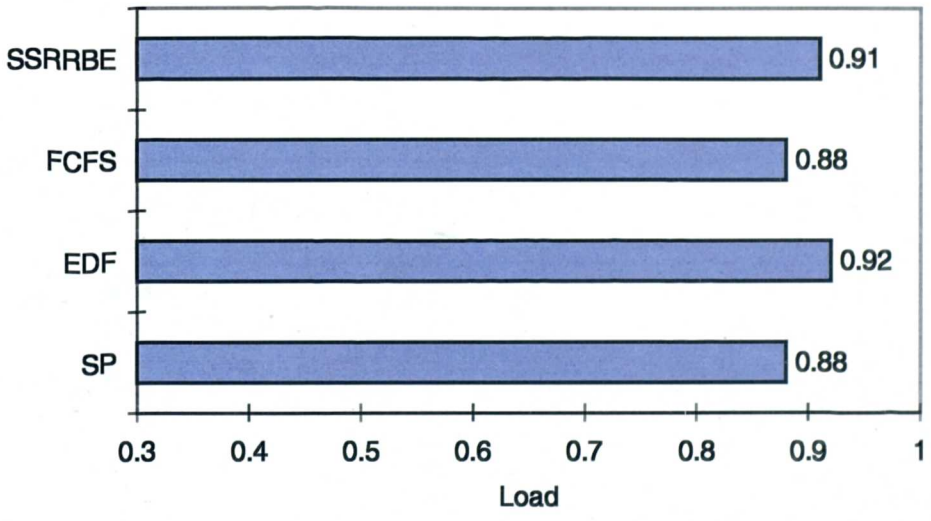

Figure 5.17:Maximum sustainable load under traffic scenario 2

5.8.2.3 Traffic Scenario 3

Since FIFO treats each packet in an equivalent manner, optimal performance would be achieved when the performance objectives of each flow are identical. In this scenario EDF reduces to FIFO since the packet with the earliest deadline is the one which is first to arrive.

Traffic Flow, i	λ_i	D_i	L_i
0	-	-	-
1	0.3	8	0.1
2	-	8	0.1

Table 5.13:Traffic scenario 3

SP performs poorly in this case since flow 2 receives much greater service than is necessary to meet the objectives and flow 1 receives what is left over. This results in a failure to meet flow 1 requirements even though flow 2 is comfortably satisfying its requirements. SSRRBE is able to distribute the probability such that both flows receive service corresponding to their offered load, thus almost emulating the performance of the FIFO discipline.

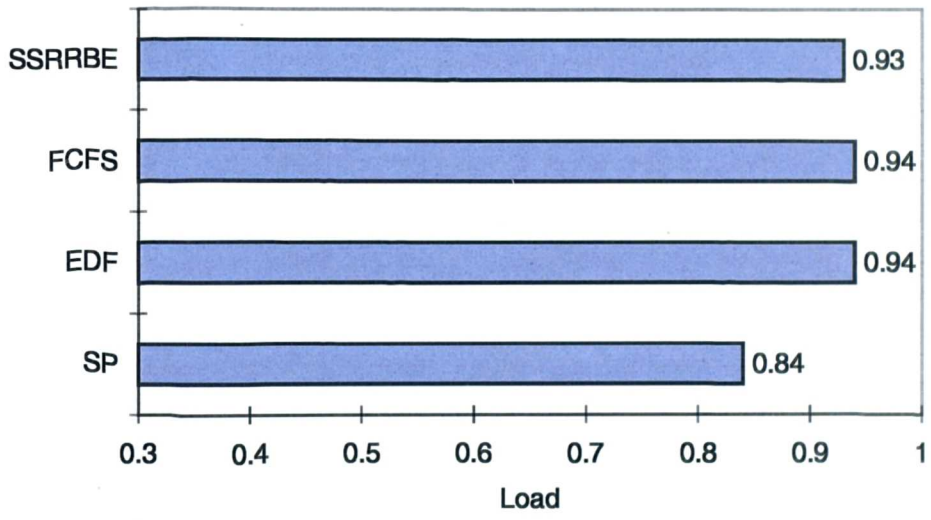

Figure 5.18:Maximum sustainable load under traffic scenario 3

5.8.2.4 Traffic Scenario 4

From the results presented we see that in each case the SSRRBE scheme is able to sustain loads within 0.01 of the best alternative algorithm. Furthermore, the best algorithm in one scenario generally does not perform well in the other scenarios. Thus we can conclude that SSRRBE is able to perform well over a wider range of traffic scenarios than the rival algorithms. We claim that the more diverse the performance objectives are then the better SSRRBE performs in comparison to the rival algorithms. To illustrate this we create an arbitrary heterogeneous traffic scenario with four traffic flows, this is summarised in Table 5.14. Starting with the arrival rates given, we increase the arrival rate of each flow uniformly. The maximum sustainable load for each algorithm is presented in Figure 5.19 and clearly the SSRRBE is able to significantly outperform the other algorithms.

Traffic Flow, i	λ_i	D_i	L_i
1	0.15+	10	0.2
2	0.15+	15	0.15
3	0.2+	6	0.1
4	0.05+	10	0.05

Table 5.14:Heterogeneous traffic scenario 4

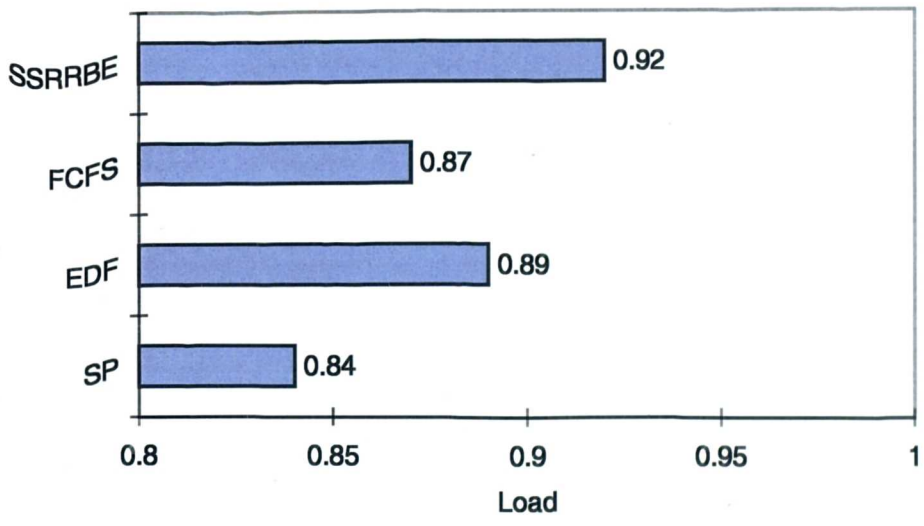

Figure 5.19:Maximum sustainable load under the heterogeneous traffic scenario 4 given in Table 5.14

The algorithms we have compared the automaton scheme with are far from comprehensive and other algorithms may slightly outperform SSRRBE in this traffic environment. However such

schemes would require a priori optimal assignment of weights or priorities to each flow, and as discussed in Chapter 2 this is a difficult problem. For example in [146] the authors suggest that the only way to successfully assign weights is through prior experimentation or simulation, such an approach is inelegant and expensive, furthermore it would be impossible to determine optimal weights for every possible traffic scenario. Effectively the proposed automaton scheme converges to these optimal weights.

5.8.3 Comparison with Guaranteed Service Disciplines

So far we have compared the learning automaton scheme with FIFO, SP and EDF scheduling algorithms, however these algorithms do not give any performance guarantees under the stochastic traffic conditions we have assumed. As mentioned in Chapter 2, in order to provide any firm deterministic performance guarantees the traffic arrival process must be deterministically bounded. In this section we compare the proposed scheme with the highly popular WFQ scheme which is capable of providing firm guarantees providing that the arrival process is leaky bucket constrained. A comparison of the maximum sustainable load of WFQ with the automaton scheme is inappropriate for two reasons. Firstly the optimal weights leading to a maximal sustainable load is not obvious, and secondly WFQ is usually applied when an admission control policy limits the carried load, thus allowing performance guarantees to be made. Instead we create a situation such that the WFQ system is just capable of providing the desired delays bounds and then demonstrate that under the same traffic conditions, the automaton scheme can satisfy these objectives and still have available capacity.

It has been proved [54] that the delay bounds offered by WFQ are within a single packet transmission time of GPS. Based upon this and assuming that the incoming link capacities are infinite in the same way as in [54], we calculate the delay bounds given the leaky bucket parameters and weights associated with each flow. Furthermore we assume that each flow is continuously transmitting according to its worst (most bursty) case, thus resulting in the highest expected delays. The traffic environment is summarised in Table 5.15

Traffic Flow, i	ϕ_i	r_i	b_i	$Max\ D_i$
1	0.6	0.5	6	11
2	0.4	0.3	6	16

Table 5.15:Traffic scenario for WFQ comparison

Since we are assuming that the flows correspond to tolerant applications we specify a maximum loss rate of 0.1 upon which the automaton probability updates are based. Figure 5.20 shows the probability associated with each queue and once convergence has taken place this results in loss rates of 0.089 and 0.084 for flows 1 and 2 respectively. It is clear that there is spare capacity in the system because of the non-zero best effort probability, therefore the system is capable of sustaining more traffic whereas the WFQ system is unable to cope with anymore traffic since the weights already sum to one. The reason for this spare capacity is twofold. Firstly, the WFQ system guarantees zero loss and since the automaton is aiming for a 0.1 loss rate it does not need to provide as much service to each of the flows. Secondly, the delay bounds under which WFQ operates are calculated based on simultaneous worst case arrival patterns which give rise to these delays, such an arrival pattern is very rare. Further improvement compared to WFQ would be apparent if it was no longer assumed that each flow transmitted according to its worst (most bursty) case, since a less bursty arrival pattern would require less probability.

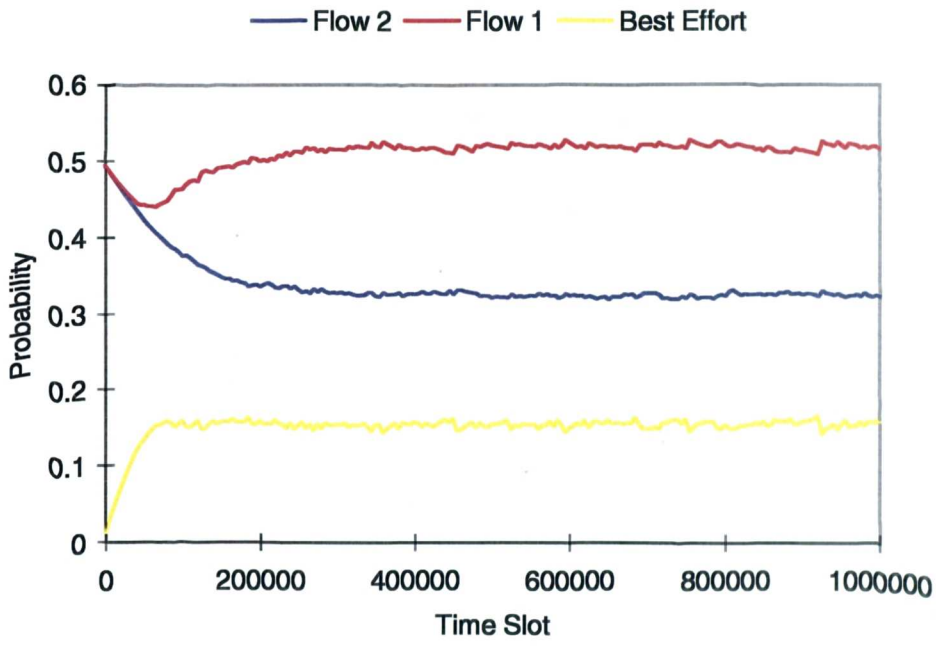

Figure 5.20:Probability plot for WFQ comparison

5.9 Summary

In this chapter we have proposed a novel scheduler aimed at supporting traffic flows with soft performance objectives which are tolerant to the occasional performance violation. The scheme is based on the use of a stochastic learning automaton with each action corresponding to the selection of a particular queue for service. The action probabilities are updated based on measurements of current performance and the aim is to provide minimal service to each traffic flow whilst still satisfying the performance objectives, thus maximising the service available for other traffic flows. The scheme is capable of supporting a heterogeneous mix of performance objectives such as a mean delay or a loss rate, where a lost packet is either a packet arriving after its deadline or a packet arriving at a full buffer.

We have shown that the automaton is able to converge to the optimal probabilities in a variety of traffic environments, including both model generated and real video traffic. One advantage of the scheme is that it does not make any prior assumptions about the traffic and is thus able to adapt should the traffic characteristics change. We have proposed an alternative to the stochastic selection of queues based on the use of frames, and for some variations on the precise selection procedure this approach can achieve higher utilisation than the stochastic approach whilst having the advantage of requiring no random number generation. We have demonstrated that the original selection procedure is flawed in that it provides more service to the best effort flow than the probability associated with it would suggest. To overcome this we proposed a variety of alternatives and concluded that procedures based on round robin excluding the best effort queue are optimal in terms of performance and complexity. The stochastic variation of this procedure has been compared to the SP, EDF and FIFO scheduling algorithms. We concluded that although it is not optimal under all traffic conditions it can provide near optimal performance under a wide variety of traffic conditions, whereas rival algorithms perform poorly in unfavourable conditions. We also concluded that the performance of the scheme improves compared to these rival algorithms as the performance objectives become more diverse. Despite this, other algorithms which involve assigning weights to each flow may still perform slightly better than the proposed scheme under such conditions, however the selection of the weights required to achieve this is difficult. We also compared the scheme with the WFQ algorithm and found that greater utilisation is possible providing that the flows can tolerate a degree of loss, which is realistic since the scheme is aimed at supporting tolerant applications. The proposed scheme is an improvement on existing measurement based queue management schemes since it is more flexible and can deal with a wider range of performance objectives, moreover allocating a probability to each flow

makes the scheme amenable to admission control whereas alternative schemes do not appear well suited to this task.

Although in this chapter we have shown that the proposed scheme is capable of outperforming rival schemes in a variety of traffic environments, such a characteristic is of no consequence in real networks unless it is possible to determine how much traffic can be supported whilst still satisfying the performance objectives. In the next chapter we introduce an admission control scheme which uses a neural network to determine the probability required for a new traffic flow and accepts the flow if sufficient probability is available.

Chapter 6

Neural Network Connection Admission Control

Accompanying any scheduling scheme there must be a corresponding admission control policy to limit the number of flows accepted onto the network, thus enabling the scheduler to perform effectively. In this chapter we attempt to formulate such an admission control policy for the scheduler proposed in the previous chapter. The scheme is very similar in nature to the class of admission control policies known as measurement based admission control policies and we begin this chapter by providing some motivation and a review of existing measurement based admission control procedures. We continue by outlining the proposed scheme which involves a neural network learning the probability that needs to be associated with a flow according to some worst case traffic descriptor. Next we present results showing the capability of neural networks to learn these required probabilities for various traffic descriptors and performance objectives. The outlined admission control scheme is then applied to a system with calls continually commencing and terminating. Its performance is compared with a more conservative counterpart together with static and adaptive WFQ under worst case and greedy traffic arrivals. Finally we present conclusions and a summary of the chapter.

6.1 Motivation

In the previous chapter we introduced a scheduling mechanism that achieves high utilisation whilst satisfying various performance objectives under a wide range of traffic conditions. However, since network resources are invariably finite, a network control policy that simply accepts all connections offered will soon result in overload, regardless of the scheduling policy used. Therefore, in order to provide specific performance objectives the network must limit the amount of carried traffic, this is achieved in the form of an admission control policy. Admission control and scheduling are, of course, related and a traffic environment which can be supported by one scheduling scheme does not necessarily mean it can be supported by another. Therefore, a

functioning admission control policy must accompany any scheduling mechanism if performance objectives are to be satisfied.

Traditional parameter based admission control relies on a priori traffic characterisations to compute the amount of bandwidth required. As discussed earlier in this thesis, such schemes often result in low utilisation for two reasons. Firstly the declared characteristics are often very conservative and represent upper bounds on the traffic arrivals since it is generally difficult to accurately estimate the traffic characteristics [147]. For example the average bit rate produced by a video codec in a teleconference strongly depends on the participants body movements, which obviously cannot be predicted with any degree of accuracy. Secondly, they often work on the assumption that traffic sources will transmit according to the worst case arrival patterns which still conform to their characterisation and again this generally results in the over allocation of resources. This type of admission control is essential when the traffic requires firm guarantees. However many applications will require soft guarantees and will be tolerant of the occasional quality of service violation. Therefore by weakening the reliability of the delay bound it is possible to achieve higher utilisation and statistical multiplexing gain. With this in mind, several authors have advocated the use of measurement-based admission control algorithms which only use the a priori source characterisations of incoming flows (or recently admitted flows) and rely on measurements to characterise existing flows. Thus, the network utilisation does not suffer significantly if the traffic descriptors are not tight.

The adaptive nature of the scheduling algorithm proposed in the previous chapter appears to have a degree of parallelism with such schemes in that it originally allocates maximum bandwidth (or alternatively probability) to new connections and then adapts to the optimal configuration based on measurements. In the next section we provide an overview of measurement based admission control.

6.2 Measurement Based Admission Control

Before going into detail about various measurement based admission control schemes we must first highlight the fact that because these approaches rely on measurement, and since source behaviour is not always static, completely reliable deterministic and statistical delay bounds cannot be provided. Thus measurement based admission control can only be applied to services with reasonably relaxed commitments.

In order to demonstrate the principle behind measurement based admission control we shall use a very simple example described in [148]. Initially consider a network scenario in which flows have a reserved rate associated with them. Let v be the sum of the reserved rates of existing flows, μ be the link bandwidth and r^i be the requested rate of the flow seeking admission to the network. A very simple non-measurement based control algorithm would simply ensure that the sum of the reserved rates and the requested rate does not exceed the available rate, therefore the flow is accepted only if the following inequality holds,

$$v + r^i < \mu \tag{7.1}$$

The main principle behind measurement based admission control is that instead of using possibly inaccurate traffic descriptors to base the admission decision on, measured values of the traffic are used. Therefore in this case, the sum of the reserved rates, v, is replaced by an on-line measurement of the aggregate rate of the existing traffic, $\hat{v}$. Thus the flow is accepted only if the following inequality holds,

$$\hat{v} + r^i < \upsilon\mu \tag{7.2}$$

The term υ is a utilisation target and prevents the system approaching full utilisation which would result in large delay variations. Furthermore, υ acts as a safeguard mechanism to protect the system against non-stationary traffic arrivals. For example, if some sources temporarily transmit at a low rate resulting in several new calls being accepted, an increase in the transmission rate of these sources may result in overload, therefore υ ensures a safety zone of spare capacity should this occur. Generally υ may be in the region of 0.9, however the precise value must be determined by the network operator and governs how conservative the network will be. In this chapter we shall be considering traffic arrivals which represent extremes of behaviour and are hence stationary, therefore for simplicity we shall assume that υ is equal to 1.

The case in which flows have requested rates is very simple and usually flows will request a performance objective given some traffic descriptor, therefore the measurement based schemes generally differ in the way measurements are taken and interpreted in order to calculate the bandwidth requirements.

The literature on measurement based admission control applied to the controlled-load and

predictive service models of IntServ as well as ATM is rapidly growing and a cross section of approaches can be found in [147][76][149][150][151][152][153][154]. Early work presented in [76] assumes certain stationary bit rate distributions and uses measurements to obtain the parameters of the distribution. This approach is obviously highly dependent on this assumption, and approaches that do not make any assumptions about the traffic characteristics are likely to be more robust.

Jamin et al. [147] suggest an algorithm for the predictive service model. This scheme assumes there exists a pseudo token bucket which describes the aggregate traffic of existing flows, the parameters of which are determined by measuring the aggregate traffic and the experienced maximal queueing delay. The token bucket parameters obtained through measurement and the token bucket parameters declared by the requesting flow are then used to calculate the required bandwidth and the maximal queueing delay of the multiplexed stream. As one would expect, if there is insufficient bandwidth or the requested maximal queueing delay is violated the flow is rejected.

The authors of [149] argue that a traffic flows rate is only meaningful if it is associated with a corresponding interval length. Based on this argument they use a rate envelope to describe the maximal traffic rate as a function of interval length, this is obtained by measuring the maximal rates over the corresponding time intervals. They introduce the concept of a schedulability confidence level which reflects the variation in past envelope measurements and describes the level of certainty that the past maximal rate envelope will continue to bound future traffic arrivals. This confidence level allows estimation of QoS parameters which are used to determine whether or not new flows should be accepted.

Floyd [150] proposes a scheme that computes the equivalent bandwidth of a set of flows using Hoeffding bounds, which are upper bounds on the tail of the distribution of a sum of random variables given the overall mean and individual maximum values of the variables. Thus the equivalent bandwidth required to satisfy a certain loss requirement is calculated from the measured average arrival rate of existing traffic and the policed peak rates of each individual flow, together with the declared token bucket parameters of the new flow. An extra feature of this scheme is that should a flow be denied admission, no other flow of a similar nature can be accepted until an existing one departs. The author deems that this approach is superior to the estimation of equivalent bandwidth assuming a normal distribution [63] as such an approach is only suitable for a large number of multiplexed flows with similar peak rates. A similar

approach is proposed in [151] in which a new call is accepted only when the measured arrival rate is below a certain threshold. The thresholds are calculated based on measurements of the aggregate arrival rate and on a single burstiness parameter p which is common to all admitted connections and represents the probability that a flow is active at a particular point in time. This assumption that p is common to all flows may prove to be a limitation of this approach, since flows with different p need to be serviced separately which will reduce multiplexing gain.

The approach from Crosby et al. [152] is based on large deviation theory and estimates the large-deviation rate function (entropy) of the traffic using traffic measurements. The entropy function allows the effective bandwidth of the existing traffic to be computed and the sum of this and the declared peak rate of the offered traffic is then used to determine whether the call is accepted.

6.3 Proposed Admission Control Algorithm

The problem of whether or not there is sufficient bandwidth to support all the traffic flows was not addressed in the discussion of the scheduling mechanism in the previous chapter and it was simply assumed that the given traffic scenario could be supported. Obviously this is a very naïve assumption since simply accepting all requests would soon result in overload of the multiplexer and the delay requirements could not be satisfied regardless of the scheduling algorithm.

6.3.1 Call Admission

The question that must be answered by an admission control scheme applied to this scheduling mechanism is whether there is enough spare *probability* that can be allocated to the new flow such that the delay requirements are met, where the spare probability is the probability currently associated with the best effort flow. Therefore we require a mechanism to estimate the required probability (service rate) to associate with a new flow, moreover, if this estimate is accurate then the convergence time of the automaton will be reduced.

Assume that a flow i requests admission to the network and provides some kind of traffic descriptor $F(i)$. Based on this traffic descriptor, the admission controller estimates the probability, p_i, that needs to be associated with this flow. A conservative approach would be to add this to the previous estimates made for the existing flows, if this sum is less than one then the flow is accepted and receives the estimated probability at the expense of the best effort traffic; otherwise it is rejected. The acceptance condition is given in (7.3).

$$\sum_{j}^{n} p_j + p_i < 1 \tag{7.3}$$

It is likely however that the estimates will be conservative since they are likely to be based on conservative traffic descriptors. Therefore, instead of using these estimates the measurement based algorithm uses the observed probabilities the automaton has converged to, $\hat{p}_j$. The acceptance condition is given in (7.4).

$$\sum_{j}^{n} \hat{p}_j + p_i < \upsilon \tag{7.4}$$

As before υ is the utilisation target and acts as a safeguard mechanism for the system. In order to prevent misbehaving sources from degrading the performance of the well behaved flows an upper limit, equal to the probability initially allocated, must be placed on the probability associated with each flow.

6.3.2 Call Termination

On the termination of a call, conventional measurement based policies do not perform any special functions since new measurements accommodate the change in conditions. However, with the proposed scheme some kind of de-allocation of probability is required such that the queues supporting these inactive flows will no longer be selected for service. The simplest method of de-allocation is to simply allocate the probability associated with the terminating call to the best effort flow. Hence if flow i terminates at time τ then the probabilities are updated as follows,

$$p_i(\tau^+) = 0 \tag{7.5}$$

$$p_0(\tau^+) = p_0(\tau^-) + p_i(\tau^-) \tag{7.6}$$

where τ^- represents the instant just before τ and τ^+ represents the instant just after τ.

6.3.3 Neural Network Probability Estimation

The limiting factor with this scheme is how to estimate the required probability based on the traffic descriptor. For relatively simple traffic descriptors it is possible to estimate the required probability using analytical models, however for more complex descriptors analytical models may

not be available or they may require a great deal of iterative computation, which may limit their real-time capabilities. For these reasons we propose to use a neural network trained using exact data (if an analytical model is computationally expensive) or data obtained from simulation (if an analytical model is unavailable) to estimate the probabilities. The function approximation capabilities of neural networks will generally allow accurate estimation, furthermore the neural network will generate a solution very quickly. Fan adopts a similar approach in [87], here a neural network is trained using data obtained from a computationally expensive analytical model to estimate the effective bandwidth of a number of multiplexed traffic sources. The neural network is able to accurately approximate the exact model but involves far less computation, thus allowing real time application. Initially we shall focus on an analytically simple scenario in which the traffic descriptor just a single parameter and then move on to a more complex realistic case.

6.4 Bernoulli traffic arrival process

Consider the simple case looked at in the previous chapter in which the arrival process at each queue is Bernoulli with a rate λ_i. If we assume the original scheduling algorithm in which the best effort queue is served if the initial queue selection is empty, then it is possible to derive analytically various delay statistics for a given probability of selection. For an arrival rate λ_i and a mean delay requirement of d_i, the required probability, p_i, is derived in Appendix D and is given by,

$$p_i = \frac{1+\lambda_i d_i}{1+d_i} \tag{7.7}$$

The surface shown in Figure 6.1 gives a graphical representation of (7.7) and the neural network must effectively learn the shape of this surface such that it can take the arrival rate and delay requirement as inputs and output the service rate. We must emphasise that in this particular case a neural network is not really necessary since the required probability can be calculated using the simple closed form solution given in (7.7), however as already mentioned this simplicity is not likely to be the general case.

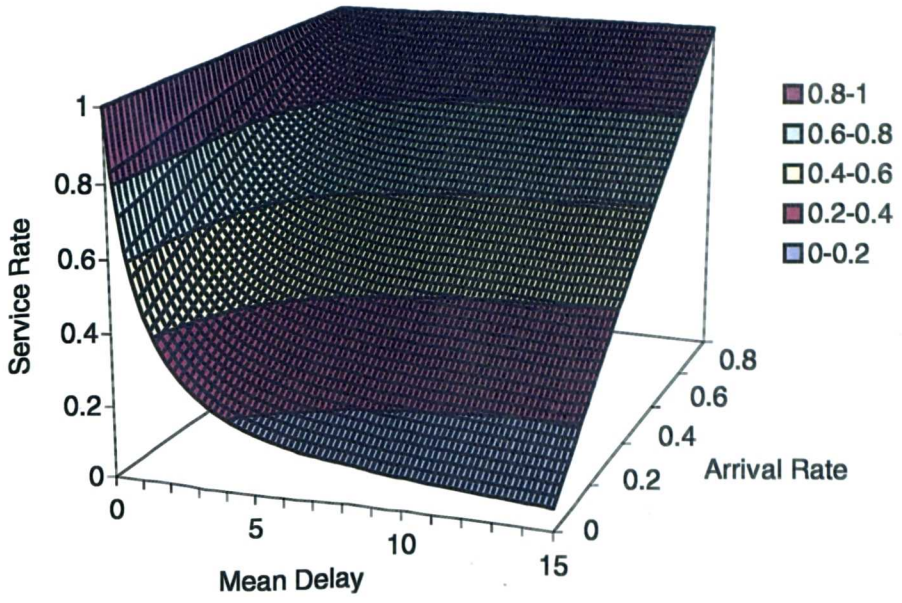

Figure 6.1:Surface showing service rate as a function of arrival rate and mean delay requirement

The training set was generated by calculating the required service rates for each combination of arrival rates in the interval [0.02, 1.0] in steps of 0.02 and delay requirements in the interval [0.2, 15] in steps of 0.2. This yielded a training set of 7425 examples. This data was presented to a neural network in a random order, and the network was trained using the back-propagation algorithm. The form of the neural network is given in Figure 6.2. The delay requirement was normalised such that it fell in the range (0,1), however the arrival rate required no pre-processing since it was already in this range.

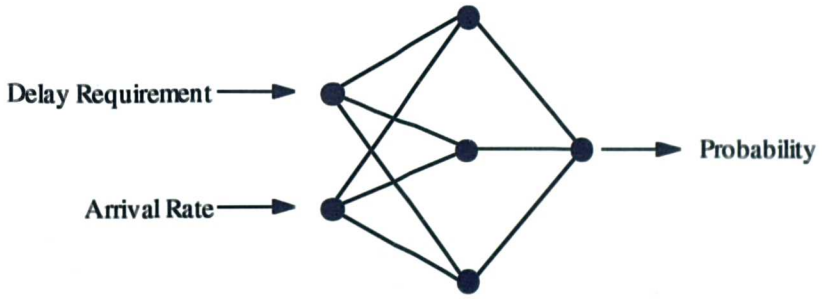

Figure 6.2: Neural network with 3 hidden neurons used to learn required probabilities

After each pass through the training set, the neural network was tested on some unseen data generated at random, this test set contained 100 examples. Figure 6.3 shows how the performance of the neural network on the test set improves as the training proceeds for various numbers of hidden neurons in a 3 layer neural network.

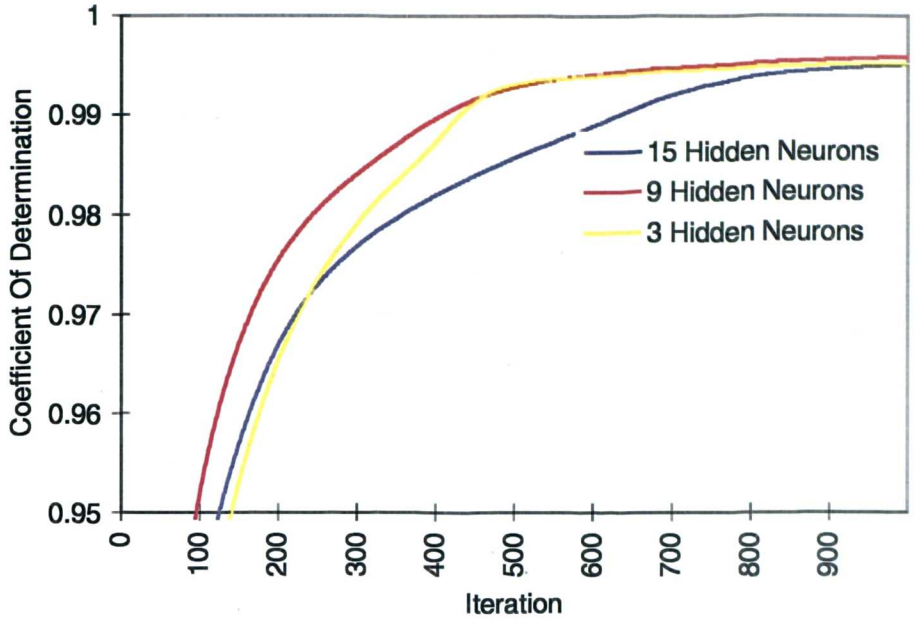

Figure 6.3:Performance of neural networks with various hidden layers on the test set

Clearly accurate estimation of the required service rate is achieved, furthermore the learning rate and final performance is similar for each neural network architecture. The performance on the test set after 1000 iterations is shown for various hidden neurons in Figure 6.4 and we find that comparable performance is achieved in each case. Since a greater number of hidden neurons increases the computational burden we conclude that in this scenario a 2-3-1 network is optimal.

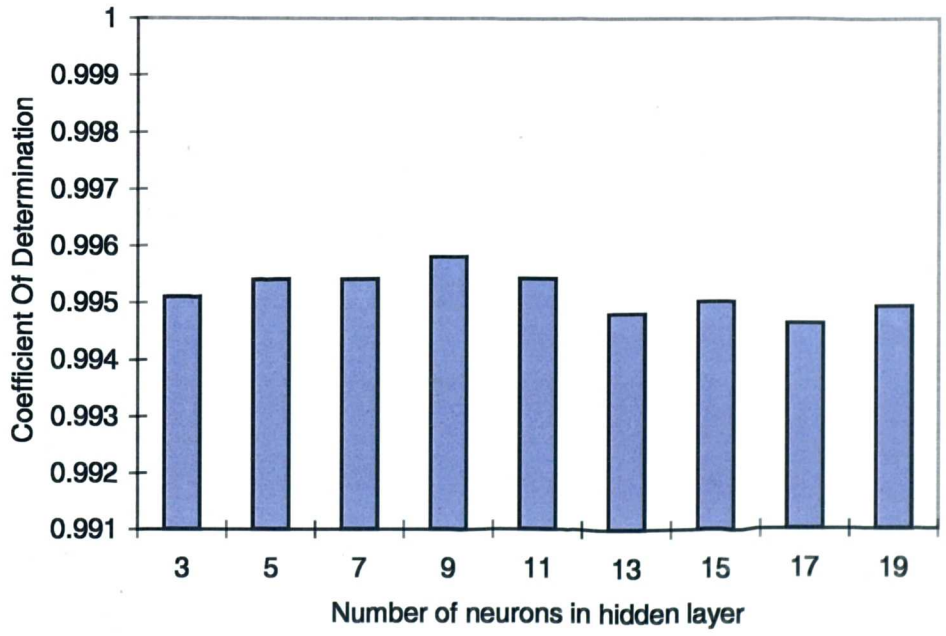

Figure 6.4:Bar chart to compare final performance of neural networks with various hidden layers on test set

We apply this neural network admission control scheme to an artificial scenario in which calls with random characteristics arrive at regular intervals and the call holding times are the same for each call. Although totally artificial, this scenario allows us to illustrate the progress of the system when calls are arriving and departing. The traffic scenario and resulting performance is summarised in Table 6.1 and Figure 6.5 shows the probability associated with each of the flows when the original scheduling with parameters of T=500, a=0.0025 and b=0.0015 is adopted. From the stability of these probabilities we conclude that the neural network is effective in estimating the required probability, furthermore the performance of each call measured over the duration of the call is satisfactory compared to the delay requirements.

Flow i	Arrival Time	λ_i	Accepted	D_i	$\hat{D}_i$
1	0	0.109	Yes	5.25	5.01
2	200000	0.106	Yes	11.82	11.08
3	400000	0.091	Yes	14.04	13.21
4	600000	0.057	Yes	12.00	11.23
5	800000	0.220	No	5.59	-

Table 6.1:Traffic scenario and performance for Bernoulli arrivals adopting the original scheduling algorithm

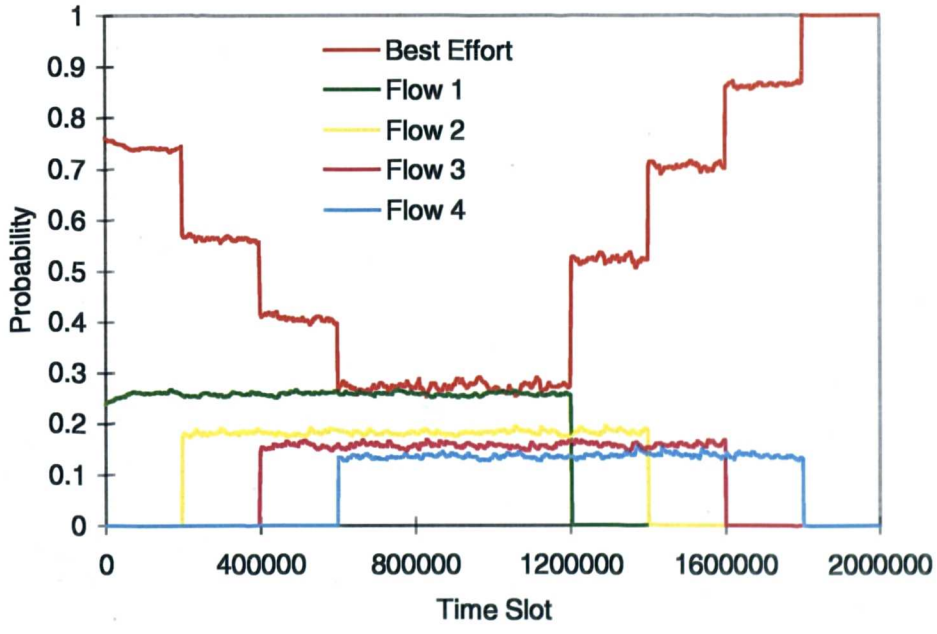

Figure 6.5:Probability plot with original scheduling algorithm under Bernoulli traffic summarised in Table 6.1

In the previous chapter we demonstrated that the original scheduling algorithm was inefficient since it served the best effort queue more frequently than the probability associated with it suggested. Based on this we concluded that the SSRRBE and FSRRBE scheduling algorithms were optimal in terms of performance and complexity. The increase in efficiency was reflected in an increased best effort probability and consequently lower probabilities for the other flows. Therefore one may argue that applying this admission control would be overly conservative when applied to SSRRBE and FSRRBE since it would estimate higher probabilities than are actually required, thus unnecessarily rejecting calls which could actually be accepted. This over estimation will often be true, however the reason why lower probabilities are required is that the round robin element of the scheme ensures additional service opportunities for the flows when the queue originally selected for service is empty. Therefore in a high load scenario in which the other queues always have something to send, the estimated probability will be accurate. In the case when the other queues are sometimes empty, the automaton would simply adapt by reducing the probability associated with that flow. The probability estimation will generally become more accurate at higher loads since the likelihood of finding non-empty queues will be greater. Moreover, it is in this region of high loads where greater accuracy is required since more calls are likely to be rejected.

Figure 6.6 shows the probability associated with each of the flows when SSRRBE is applied to the same traffic scenario as summarised in Table 6.1 and the resulting performance is given in Table 6.2. From Figure 6.6 it is clear that the SSRRBE scheme is capable of sustaining a greater load than the original scheduling algorithm and upon an arrival or departure of a call the probabilities are re-distributed such that the performance objectives continue to be satisfied. The conservative probability estimation results in the performance being further from requested compared to the original scheduling algorithm, this is because the automaton takes longer to converge, however the longer the call holding times then the less noticeable this will be.

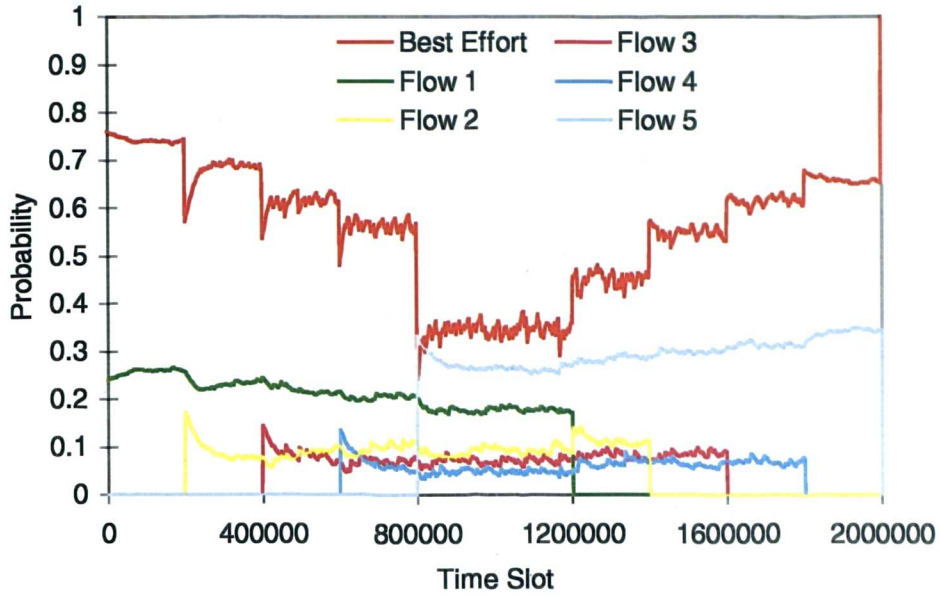

Figure 6.6: Probability plot with SSRRBE scheduling algorithm under Bernoulli traffic summarised in Table 6.2

Flow i	Arrival Time	λ_i	Accepted	D_i	$\hat{D}_i$
1	0	0.109	Yes	5.25	4.84
2	200000	0.106	Yes	11.82	10.45
3	400000	0.091	Yes	14.04	12.37
4	600000	0.057	Yes	12.00	10.68
5	800000	0.220	Yes	5.59	5.17

Table 6.2: Traffic scenario and performance for Bernoulli arrivals adopting the SSRRBE scheduling algorithm

6.5 Token bucket constrained traffic arrival process

The Bernoulli traffic descriptor applied in the previous section provided a basis for generating a training set analytically, however such a traffic descriptor is unrealistic and a more applicable one is the token bucket descriptor described in [54]. The traffic is specified by three parameters (r, b, C) : r is the rate at which tokens arrive at the bucket, b is the depth of the token bucket and C is the maximum rate at which packets can leave the bucket. These restrict the traffic in terms of the average sustainable rate (r), peak rate (C) and burstiness (b and C). Using these parameters it is possible to calculate a worst case traffic arrival process that would yield the greatest delays. Such an arrival process occurs when the tokens from a full bucket are consumed as quickly as possible resulting in packets transmitted at the peak rate C. Once the bucket

becomes empty it is allowed to refill and once full the process repeats. Therefore the worst case traffic scenario is an interrupted deterministic process with fixed length active and idle periods.

Assume that the bucket depth is continuous and that each packet requires a single token. If a packet is transmitted at t=0 then for an original bucket level b_0, the level of the bucket, $L(t)$, is given by,

$$L(t) = (b_0 - 1) - Ct + rt \tag{7.8}$$

This equation arises from the fact that the bucket is being drained at a rate of C and filled at a rate of r. In order to find the maximum active time we must evaluate the time when the bucket becomes empty given that the original bucket level corresponds to a full bucket b. The solution is given by substituting $L(t)$=0 and b_0=b into (7.8) and rearranging,

$$t = \frac{b-1}{C-r} \tag{7.9}$$

Now in this period only a finite number of packets can be transmitted, therefore the actual number of packets transmitted in this period is given by,

$$N = \left\lfloor \left(\frac{b-1}{C-r} \right) C \right\rfloor + 1 \tag{7.10}$$

(N.B The plus one term arises since a packet is transmitted at time t=0). The time at which the N^{th} packet is transmitted is given by,

$$T_{ON} = \frac{\left\lfloor \left(\frac{b-1}{C-r} \right) C \right\rfloor}{C} \tag{7.11}$$

Now at this time the bucket level is given by,

$$L(T_{ON}) = b - N + rT_{ON} \tag{7.12}$$

Therefore assuming no additional packet arrivals and the bucket simply fills at the rate of r, then

the time it takes for the bucket to become full again is given by,

$$T_{OFF} = \frac{b - L(T_{ON})}{r} = \frac{N - rT_{ON}}{r} \tag{7.13}$$

So the worst case arrival process for a token bucket constrained source with parameters (r, b, C) is an interrupted deterministic process with active and idle times given in equations (7.11) and (7.13) respectively and inter-arrival times during the active period given by C.

The solution of a queueing system with an interrupted deterministic arrival process and a geometric server is complicated and to the best of our knowledge an accurate solution does not exist [155]. Therefore the behaviour of such a queue must be derived empirically through simulation and the results used to train the neural network. Even if a solution is available analytically it is likely to involve significant computation, therefore using a trained neural network will allow solutions to be obtained much faster.

6.5.1 Mean Delay Requirements

Assuming that the performance objective is in terms of a mean delay requirement the neural network must effectively learn the function that maps the token bucket parameters and the delay requirement to the required probability, this function is given in (7.14)

$$p_i = \hat{F}(r_i, b_i, C_i, D_i) \tag{7.14}$$

For a variety of leaky bucket parameters and service rates, the resulting mean delays were obtained through simulation such that the 90% confidence interval was within 1% of the result. The training set contained 1226 data points. This data was presented to the neural network in a random order and the neural network was trained using the back propagation algorithm with a learning coefficient of 0.1. After each pass through the training set, the performance of the neural network was tested on some previously unseen data which had been generated in the same manner as the training set. The test set contained 183 data points. Figure 6.7 shows how the neural network performs when evaluated using the test set and clearly the neural network with 3 hidden layer neurons performs worse than those with more hidden layers. However, the performance of the neural networks with more hidden neurons is similar, therefore the accuracy/complexity trade-off would suggest that 5 hidden neurons is most suitable.

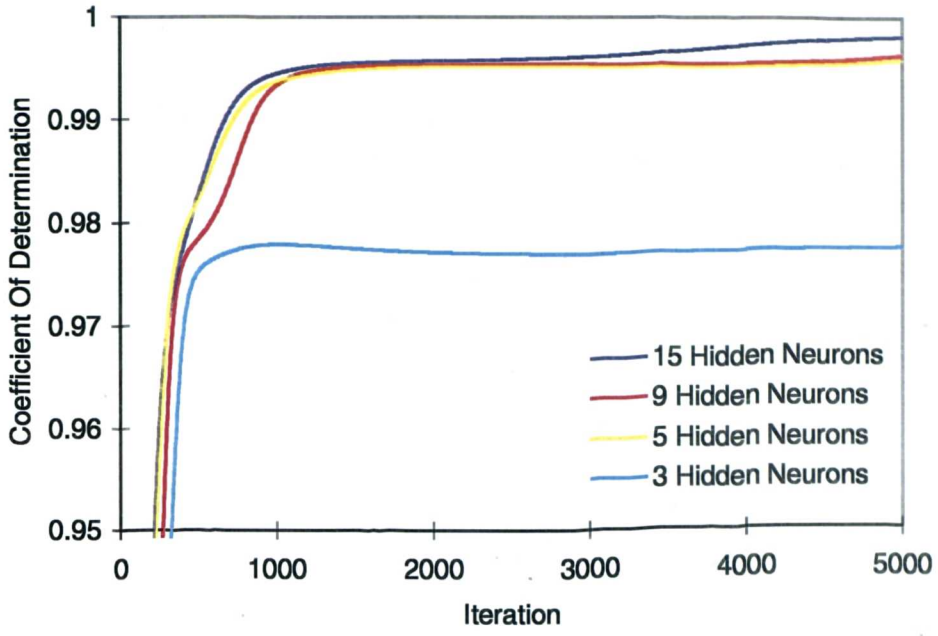

Figure 6.7:Performance of neural networks with various hidden layer neurons on the test set

In order to illustrate the accuracy of the neural network estimation we show the probability plot when calls with random traffic characteristics obeying the worst case arrival process arrive at regular intervals and the original scheduling algorithm is adopted at the multiplexer. The probability plot is given in Figure 6.8 and a summary of the traffic and performance is given in Table 6.3.

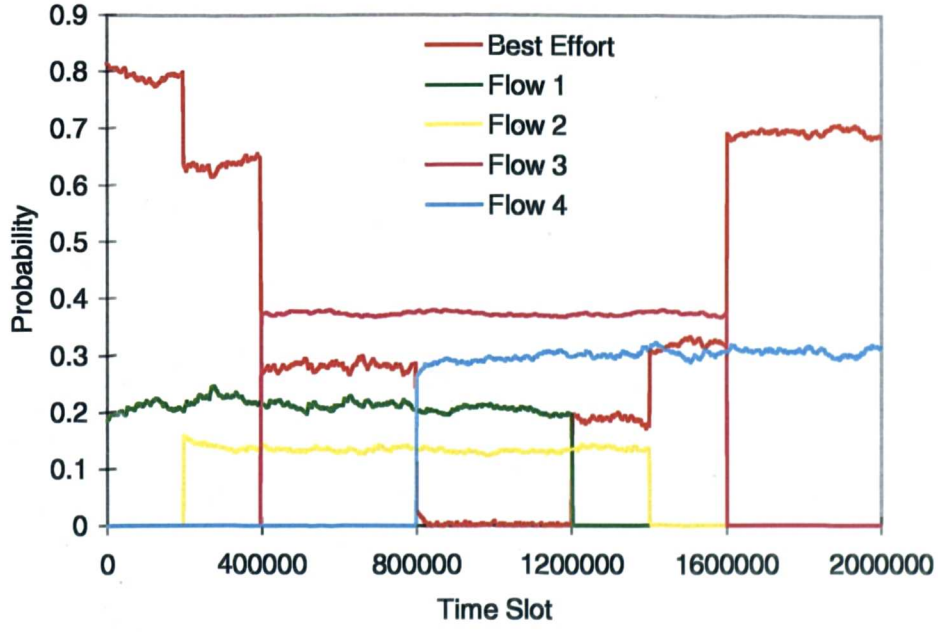

Figure 6.8:Probability plot with original scheduling algorithm under worst case traffic summarised in Table 6.3

As expected the probabilities remain relatively constant throughout the duration of the calls and the performance observed by each call is satisfactory.

Flow, i	Arrival Time	r_i	b_i	C_i	Accepted	D_i	$\hat{D}_i$
1	0	0.011	6.82	0.157	Yes	6.81	6.28
2	200000	0.027	6.66	0.069	Yes	8.87	8.35
3	400000	0.127	7.97	0.692	Yes	7.63	7.37
4	600000	0.082	6.67	0.965	No	7.32	-
5	800000	0.024	8.93	0.384	Yes	7.38	6.88

Table 6.3:Traffic scenario and performance summary for worst case traffic sources with original scheduling algorithm

Assuming worst case arrivals is often overly conservative and next we consider the case in which the traffic sources are greedy [54] and continuously attempt to send the maximum amount of traffic. Greedy sources experience lower delays than worst case sources under identical token bucket parameters, since the former do not save up bursts. Adopting the same token bucket parameters as in the worst case scenario, Figure 6.9 shows the probability associated with each flow and in each case a probability lower than estimated is required.

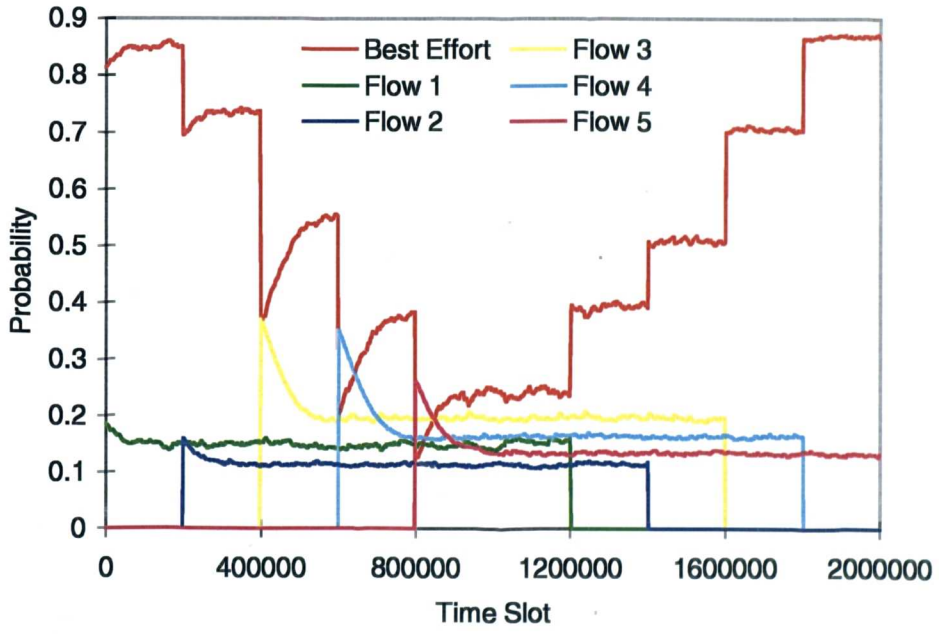

Figure 6.9:Probability plot with original scheduling algorithm under greedy traffic summarised in Table 6.4

Applying the SSRRBE scheduling algorithm to the same scenario yields the probability plot given in Figure 6.10 and the performance of both the original and SSRRBE scheduling algorithms are summarised in Table 6.4. In each case the performance requirements are satisfied. An interesting point regarding Figure 6.10 is that the probability associated with flow 2 occasionally falls to the minimum probability, which we have arbitrarily chosen to be 0.01. The reason for this is that this flow is not very demanding and it receives sufficient service from the round robin default action which is activated whenever an empty queue is selected. If the minimum probability was zero then should the other calls terminate, this flow would receive no service indefinitely. By contrast, enforcing a non-zero minimum probability ensures some service for this flow, and although initially this service would be insufficient it allows the automaton to adapt.

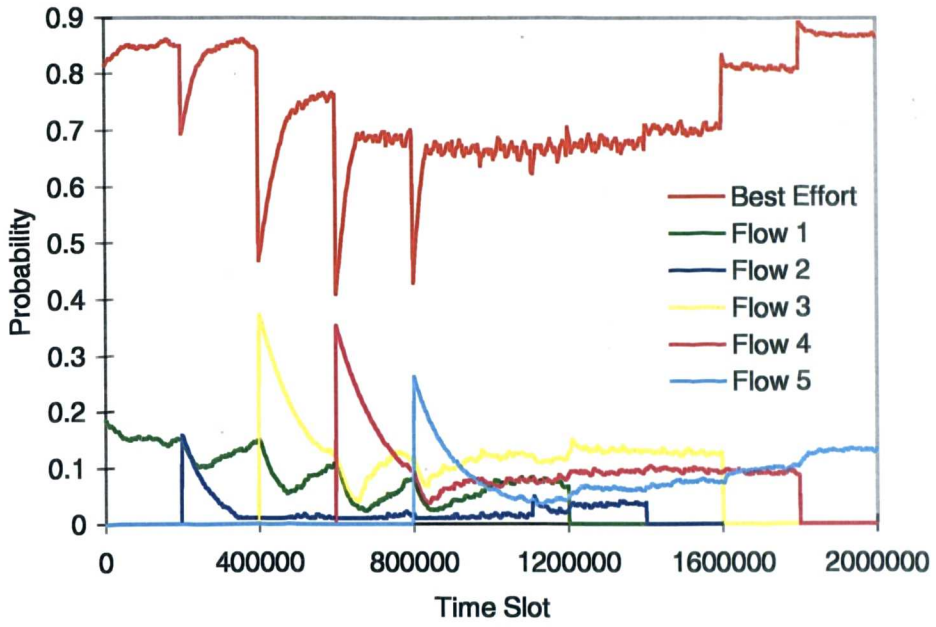

Figure 6.10:Probability plot with SSRRBE scheduling algorithm under greedy traffic summarised in Table 6.4

Flow, i	Arrival Time	r_i	b_i	C_i	D_i	$\hat{D}_i$ (SS)	$\hat{D}_i$ (SSRRBE)
1	0	0.011	6.82	0.157	6.81	6.19	6.00
2	200000	0.027	6.66	0.069	8.87	8.30	7.24
3	400000	0.127	7.97	0.692	7.63	6.83	6.51
4	600000	0.082	6.67	0.965	7.32	6.60	6.32
5	800000	0.024	8.93	0.384	7.38	6.74	6.64

Table 6.4: Traffic scenario and performance summary for greedy traffic sources with original and SSRRBE scheduling algorithms

6.5.2 Maximum Delay Requirements

So far we have assumed that the performance objective associated with the calls requesting admission is simply a mean delay. In the previous chapter we showed that a number of performance objectives may be satisfied, including maximum delay requirements. For the case of Bernoulli arrivals it is quite simple to extend the neural network admission control scheme to accommodate maximum delays by simply training a neural network on an analytically generated data set such that it maps the arrival rate, maximum delay requirement and tolerated loss rate to the required probability. For the token bucket constrained traffic we may generate empirically another training set and use this to train another neural network. However, since we must simulate various combinations of maximum delay and tolerable loss rate together with various token bucket parameters, the training set generation would be computationally expensive. Instead we remember that providing a queue is served according to a Bernoulli process the delay distribution is geometric, regardless of the arrival pattern [145]. Therefore it is possible to calculate the mean delay from the maximum delay requirement and tolerable loss rate and then apply the previously trained neural network to perform the admission control. Effectively the maximum delay requirement, x, is given by the $(1\text{-}L)^{th}$ percentile of a geometric distribution of mean m, where L is the tolerable loss rate.

The function that relates these parameters is given by,

$$m = \frac{1}{1+\exp\left(\frac{\ln L}{x+1}\right)} - 1 \qquad (7.15)$$

Given the traffic scenario summarised in Table 6.5 we apply this admission control scheme assuming worst case arrivals and the original scheduling algorithm, thus the estimated probabilities should be equal to those actually required to satisfy the objectives. The probability plot is given in Figure 6.11 and clearly accurate probability estimation is achieved, furthermore the performance objectives associated with each flow are satisfied (Table 6.5)

Flow	Arrival Time	r_i	b_i	C_i	D_i	L_i	$\hat{L}_i$
1	0	0.180	8.91	0.502	18.6	0.144	0.130
2	200000	0.060	3.88	0.648	12.4	0.044	0.037
3	400000	0.076	4.74	0.476	20.4	0.061	0.053

Table 6.5: Traffic scenario and performance summary for worst case traffic sources with original scheduling algorithm satisfying maximum delays

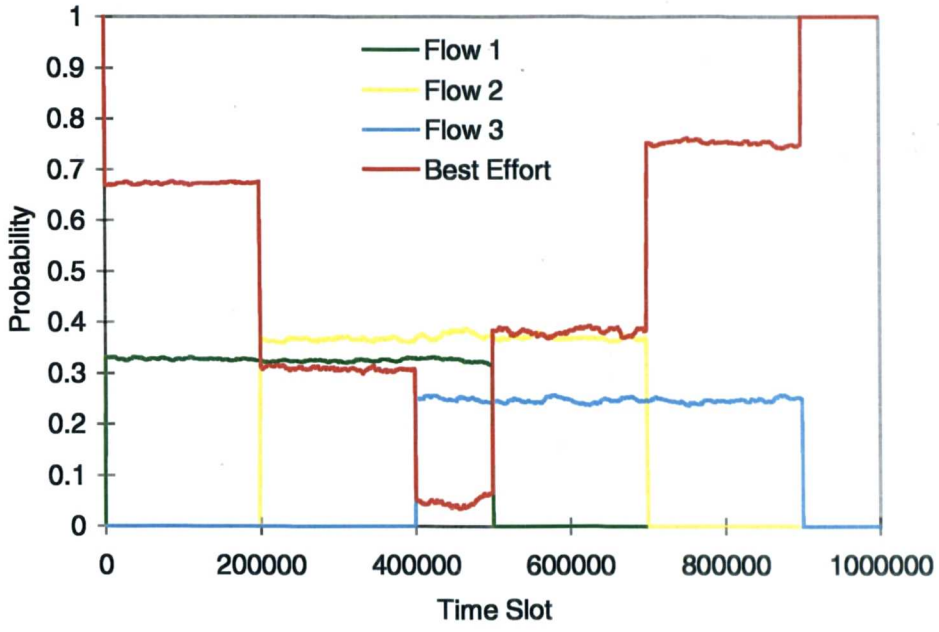

Figure 6.11: Probability plot with original scheduling algorithm under worst case traffic summarised in Table 6.5

6.6 Comparison with Alternative Schemes

Since the scheduling algorithm is inherently linked to the admission control scheme, it is impossible to compare performance with other proposed measurement based procedures. Initially we shall compare the scheme with a conservative non-measurement based approach. We continue by comparing with a static WFQ scheme which can mathematically guarantee the delay bounds requested by the sources and an adaptive WFQ scheme based on the same principles as the proposed scheme.

The call model we adopt is one commonly used in the literature in which call requests arrive according to a Poisson process and the holding times of each call are exponentially distributed [156]. The mean call holding time was chosen to be 500,000 time slots and the mean call inter-arrival time was 50,000. This factor of 10 ratio between the two parameters ensures that the system is continually faced with new calls, this ensures that the system remains close to maximum utilisation and that there is a reasonable turnover of calls. This choice of parameters is equivalent to a situation in which the link capacities are 25 Mbit/s, the average arrival rate of calls is 1 call/s and the average call holding time is 10s. The token bucket parameters and delay requirements associated with each call are generated at random, the maximum peak rate of each call is the link rate of 25 Mbit/s, however the maximum mean rate it limited to 5 Mbit/s such that a reasonable number of calls can be supported.

6.6.1 Comparison With Conservative Scheme

The conservative policy is one in which calls are admitted based on the previous estimates of the required probabilities and the probabilities are not updated. The acceptance condition for this approach is given in equation (7.3).

6.6.1.1 Worst Case Traffic Sources

Initially we shall focus on the situation in which the calls are modelled with worst case arrivals and we shall assume that the control parameters associated with the scheme are T=500, a=0.0025 and b=0.0015. Both admission control policies are exposed to identical random traffic arrivals and we find that the mean number of active calls with the conservative admission control policy is 3.00, by contrast the mean number of active calls for the proposed measurement based policy is 5.54, the corresponding distributions are given in Figure 6.12. In each case 1936 calls requested admission, however 32% were accepted by the conservative scheme whereas 57% were accepted by the measurement based scheme and in each case all accepted flows met their performance objectives. From this we can conclude that the measurement based scheme results in greater network utilisation than the conservative scheme and is still capable of satisfying the performance objectives.

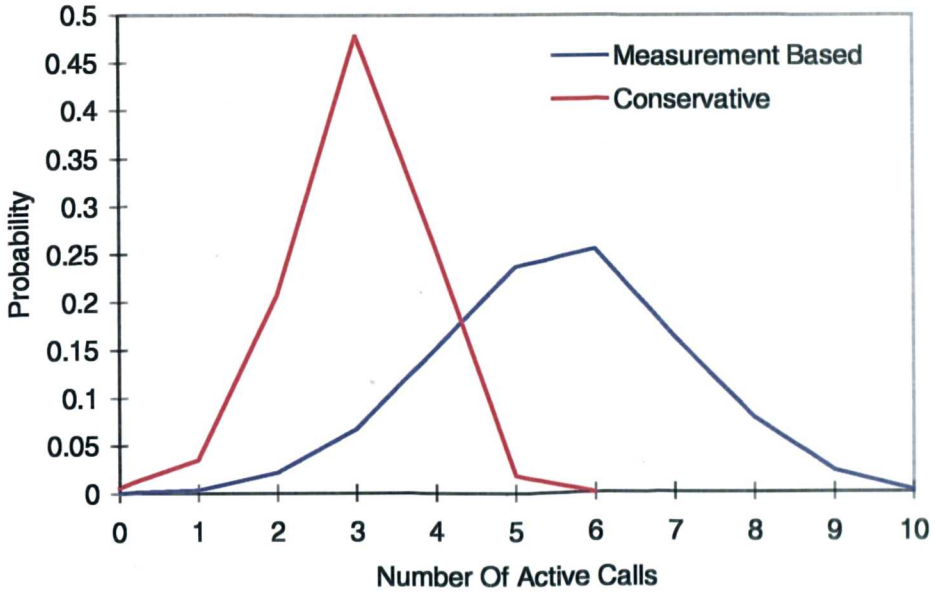

Figure 6.12:Distribution of number of active calls under worst case traffic sources

6.6.1.2 Greedy Traffic Sources

For both the conservative and measurement based schemes, the admission control policy assumes worst case arrivals given the token bucket parameters. However, as previously mentioned it is often the case that the actual arrival process may be quite different from the worst case resulting in lower delays given the same service rates. In this section we adopt the same model as in the previous section except that we assume greedy traffic sources and as already demonstrated, such sources require less probability than the corresponding worst case sources, hence resulting in greater available capacity. Therefore one would expect that the difference in performance between the conservative and measurement based schemes to be greater than with worst case arrivals.

Figure 6.13 shows the distribution of the number of active calls for the conservative and measurement based schemes and these lead to means of 3.00 and 6.10 respectively. Out of the 1936 calls requesting admission the conservative scheme accepts 32% whilst the measurement based scheme accepts 62%, and in both cases the performance objectives of each call are satisfied. Therefore as expected the difference in performance between the proposed scheme and the conservative case is more pronounced under these conditions. In reality, the actual arrival process will fall somewhere between worst case and greedy and the performance improvement of the proposed measurement based scheme compared to the conservative scheme will depend upon this arrival process.

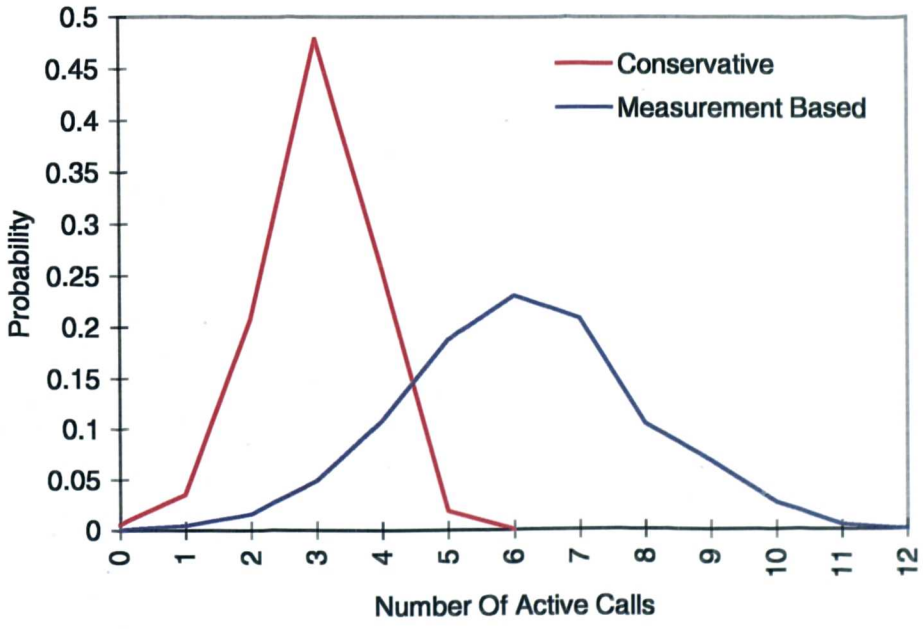

Figure 6.13:Distribution of number of active calls under greedy traffic sources

6.6.2 Comparison with WFQ

In this section we compare the scheme with the admission control scheme associated with WFQ. This scheme allocates a weight to each flow such that the maximum delay requirement can be mathematically guaranteed providing that the source remains token bucket constrained. One would expect that the proposed measurement based scheme would outperform WFQ since it allows for a specified fraction of late packets whereas WFQ allows no late packets. Moreover the time stamping and real-time priority sorting makes the WFQ scheduling algorithm computationally demanding, such real time computation is not required in the proposed scheme since the scheduling algorithm is just based upon probability.

We also compare the scheme with a modified WFQ algorithm in which the weights, which represent the fraction of the outgoing link capacity allocated to each flow, are no longer static and can be updated in exactly the same way as the probabilities in the proposed scheme. The admission decision is similar to that of the automaton scheme in that the call is accepted if a sufficient fraction of the outgoing link capacity has not been allocated. The acceptance condition is given by,

$$\sum_{j}^{n} \hat{\phi}_j + \phi_i < \upsilon \tag{7.16}$$

where ϕ_i is weight associated with flow i. One might expect such a scheme to outperform static WFQ and the automaton based scheme since it inherits the advantages from both schemes. From the automaton based scheme it inherits the property of allowing the occasional late packet to increase utilisation and from the WFQ it inherits a sophisticated scheduling algorithm.

Since in these comparisons we need to measure small loss rates, we double the measurement period to T=1000 and for consistency we double the reward and penalty parameters such that a=0.005 and b=0.003.

6.6.2.1 Worst Case Traffic Sources

Once again we assume that the sources are transmitting at their most bursty whilst still being token bucket constrained. The distribution of the active number of calls is given in Figure 6.14 and this corresponds to a mean number of active calls of 3.39, 5.26 and 6.86 for the static WFQ, automaton and adaptive WFQ schemes respectively.

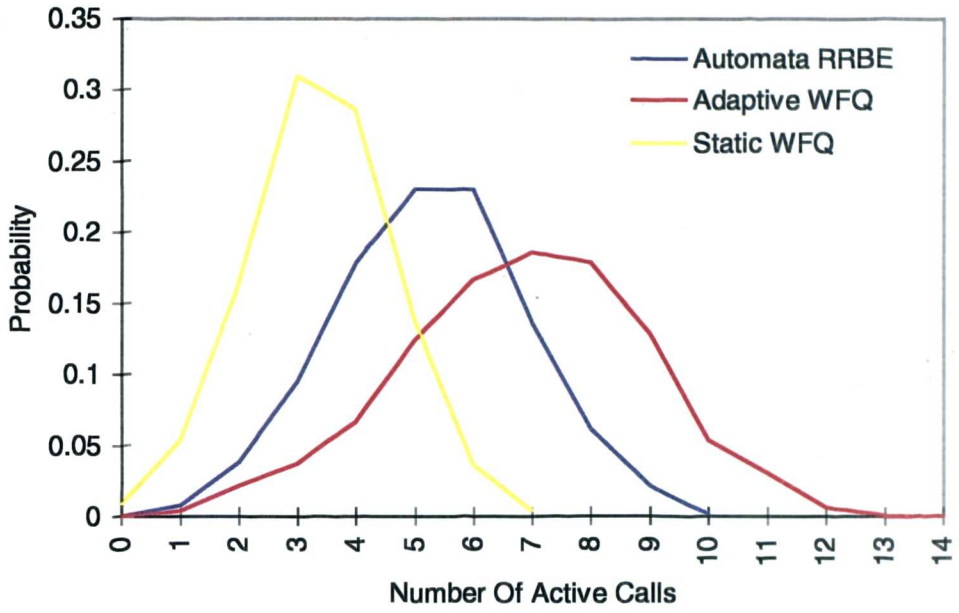

Figure 6.14:Distribution of number of active calls under worst case traffic sources

Of the 1936 calls requesting admission the static WFQ accepts 37%, the automaton scheme accepts 56% and the adaptive WFQ accepts 71%. Clearly the adaptive WFQ results in the highest utilisation, however of the calls accepted by this scheme 16% fail to meet their performance objectives whereas with the other schemes all accepted calls meet their respective objectives. Obviously such a high percentage is unsatisfactory and the reason for this is that the rate based nature of the WFQ scheme is such that any spare capacity is shared amongst the active flows in proportion to their weights. Therefore the weights associated with each flow only represent a lower bound on the fraction of the available bandwidth it can receive. Thus the sum of the weights is often much less than the aggregate bandwidth the flows are actually receiving. Using this sum leads to an overestimate of the spare capacity resulting in accepted calls which cannot be supported. For the automaton scheme, although the probability associated with each individual flow may not accurately represent the service it receives, the sum of the probabilities associated with each flow is at least as great as the aggregate service received, thus allowing accurate estimations of the available capacity.

6.6.2.2 Greedy Traffic Sources

One would expect the performance differential between the adaptive schemes and static schemes to increase when the traffic sources no longer exhibit worst case behaviour. A greedy traffic scenario yields the active call distributions given in Figure 6.15 and results in a mean number of active calls of 3.39, 5.95 and 6.97 for the static WFQ, automaton and adaptive WFQ schemes respectively. In each case 1936 calls request admission, the static WFQ scheme accepts 37% of these, the automaton scheme accepts 62% and the adaptive WFQ scheme accepts 72%. As expected the adaptive schemes accept more calls in this scenario compared to worst case traffic conditions. Once again however, although the adaptive WFQ scheme results in the highest utilisation this is at the expense of 23% of the accepted calls failing to meet their performance requirements compared to all of the calls meeting their objectives with the static WFQ and automaton schemes.

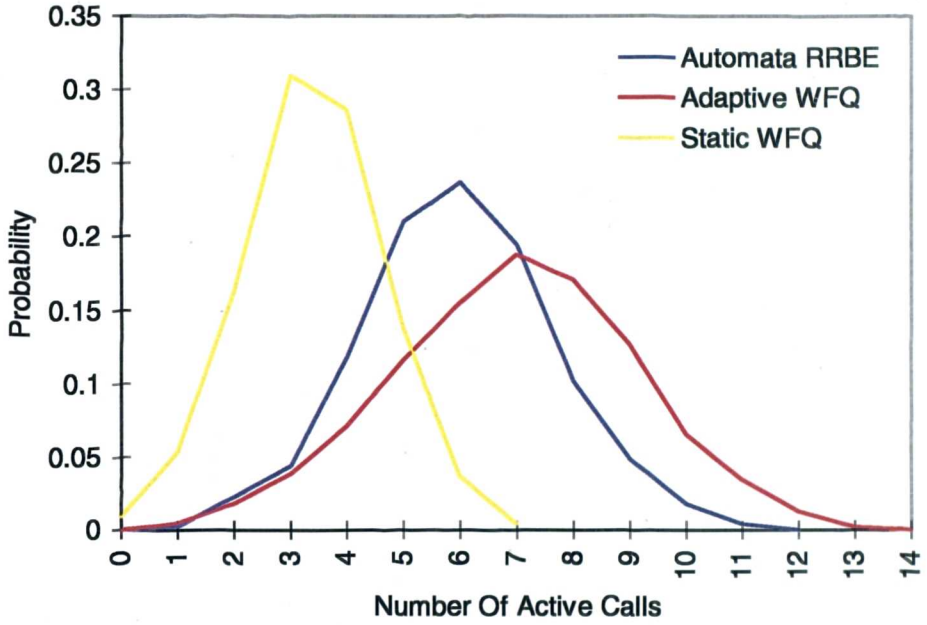

Figure 6.15:Distribution of number of active calls under greedy traffic sources

6.6.3 Discussion

The previous comparisons have all been made under traffic conditions in which the mean call arrival rate and mean call holding times are constant. The performance of the adaptive schemes depend upon the length of the call holding times, more precisely they depend upon the ratio of the call holding times to the measurement interval and reward and penalty parameters. If the call holding times are long, this allows the adaptive schemes to converge to the optimal probabilities and thus maximise the spare probability which can then be used to accept new calls. By contrast,

short call holding times prevent the adaptive schemes from converging and we observe performance closer to the static schemes. Such a characteristic is crucial with any adaptive control scheme since for any adaptive scheme to be effective the conditions must remain steady for long enough periods such that the scheme can adjust and be of some benefit. An adaptive scheme which does not have sufficient time to adapt is simply a static scheme.

The control parameters T, a and b also influence the performance of the scheme, more precisely it is the ratio of these parameters that governs performance. If a and b are too small compared to T then convergence takes place very slowly and we find that a fewer number of calls are accepted, since the excess probability associated with the flows is not made available fast enough. Furthermore, in this case some of the accepted calls may fail to satisfy their objectives since the probability associated with a call is not increased quickly enough when it is experiencing unsatisfactory performance. If a and b are too large compared to T then the probabilities tend to oscillate and this sometimes leads to the situation where the instantaneous probability distribution indicates that a call can be accepted when in fact it cannot. The opposite case may also occur in which the instantaneous probabilities indicate that a call cannot be accepted when in fact in can. Thus if a and b are too high then we observe fewer accepted calls and more calls failing to satisfy their requirements. Figure 6.16 illustrates these properties and shows the number of accepted calls and the number of these calls which satisfy their performance objectives for various values of the reward parameter a, in each case the penalty parameter b is 3/5 of a and the measurement interval T is 500..

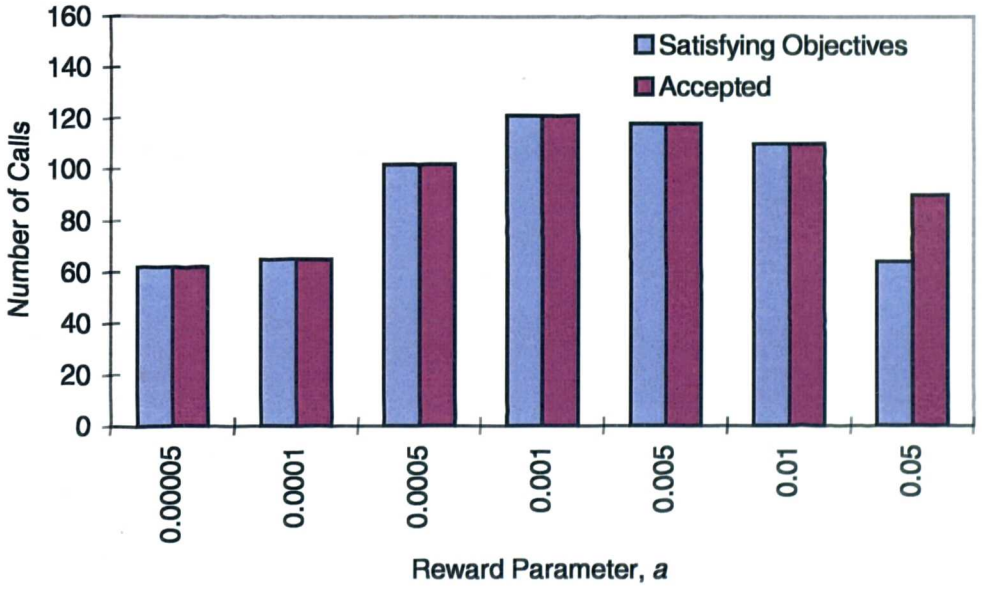

Figure 6.16:Bar chart to indicate performance of scheme with various values of a

Although these parameters influence performance it would be quite easy to tune these since they do not depend heavily on the individual traffic characteristics and requirements, this differs from other schemes such as WRR in which the requirements of the traffic govern the parameters.

6.7 Summary

In this chapter we have proposed an admission control policy to accompany the scheduling mechanism outlined in the previous chapter. The scheme is based on a neural network estimating the probability that needs to be associated with a new flow and comparing that with the current unallocated probability in the scheduler. Initially the scheme was applied to a very simple situation in which the probabilities could be derived analytically and we demonstrated that the neural network was capable of learning the correct probabilities. We highlighted that such simple analytically tractable scenarios may not be commonplace and that more sophisticated traffic descriptors may lead to intractable queueing analysis or to situations in which the analysis requires a great deal of computation, which would inevitably limit the real-time application of the scheme. The worst case arrival pattern of a traffic source characterised by a token bucket falls into the intractable category and using results obtained from simulation a neural network was trained to learn the required probabilities given the traffic descriptor and requirements. We demonstrated that the neural network was able to accurately learn the required probabilities and we continued by showing that the proposed scheme was able to achieve greater utilisation compared to a more conservative non-measurement based scheme. Next we provided a comparison with the static WFQ scheme and demonstrated that the proposed scheme could support more calls whilst still satisfying the performance objectives. An adaptive modification to WFQ was also outlined and compared with the proposed scheme, and although more calls were supported by the adaptive WFQ scheme the rate based nature of the scheduler lead to a significant fraction of accepted calls failing to meet their objectives.

Chapter 7

Conclusions and Further Work

In this thesis we have investigated the potential of learning algorithms applied to the problem of traffic management and congestion control in integrated services networks. Integrated services networks will be based on the paradigm that before communication starts the source must declare its traffic characteristics and based on this declaration an admission controller must choose to accept or reject the connection. If the connection is accepted then appropriate network resources must be allocated and some kind of commitment as to the quality of service received must be made providing that the source conforms to its traffic characteristics.

The fundamental problem with congestion control is that of traffic characterisation. In order to carry out effective congestion control and achieve high network utilisation it is necessary for the source to accurately describe its traffic characteristics. Accurate traffic characterisation requires a number of parameters which may be difficult to estimate and police and may lead to complex resource allocation calculations. A simple set of parameters would be more straightforward to estimate and police, and would generally lead to simple calculations regarding resource allocation. The problem is that this would often lead to poor utilisation since any resource allocation would be conservative since it must be based on a worst case arrival pattern which conforms to the simple traffic characterisation. In order to achieve higher utilisation it may be necessary to adopt measurement based policies which rely on the prediction of future resource requirements based on current traffic measurements. However the dynamic nature of network traffic may lead to the occasional performance violation and therefore only soft performance guarantees can be made. The learning and adaptation capabilities of some artificial intelligence techniques make them amenable to this problem and we have investigated the possible use of neural networks for performing traffic prediction and also the use of stochastic learning automata to perform adaptive scheduling in order to maximise utilisation. Other characteristics such as high computational rates and the lack of a requirement for mathematical models reinforces the argument that artificial intelligence could be used for some aspects of network control and to

illustrate this we have used a neural network to learn an unknown function in an admission control policy.

The thesis introduction defined integrated services networks and described various applications which will be supported on such networks. In particular we identified applications which require deterministic and statistical performance guarantees which may either be firm or soft guarantees. Various key aspects of an integrated services architecture, including the service interface, service commitments and congestion control were outlined and we described the two most popular integrated services network architectures, namely ATM and Intserv. In Chapter 2 we presented a state-of-the-art survey of traffic management and congestion control techniques, the aspects reviewed included traffic enforcement, queue management, admission control and reactive control. We identified that the effectiveness of any scheme is directly related to its complexity and the sophistication of the traffic model used. In the next chapter we reviewed various applications of artificial intelligence to the problem of network control, the techniques examined included neural networks, fuzzy logic and stochastic learning automata. A number of possible areas suitable for further investigation were identified, these included the application of neural networks to perform traffic prediction, the use of stochastic learning automata to control queueing systems with unknown parameters and the use of neural networks to learn functions for which mathematical models do not exist or are too computationally demanding. In Chapter 4 we investigated the potential of using neural networks to perform traffic prediction with the goal being for the neural network to learn the correlations which exist in network traffic. Contrary to popular belief we reached the rather intuitive conclusion that generally the prediction of real network traffic is impossible and that the recent successful applications of neural networks to perform predictions were due to the fact that mathematical traffic models were used or the traffic exhibited periodic properties. In these cases the neural networks were able to learn the mathematical generating function or learn the periodicity of the trace. We found that generally the neural network learned that the best prediction results were obtained by tracking the actual traffic trace. In such cases the current observation was used as a prediction for the next observation, this is analogous to the weatherman who predicts tomorrow's weather to be the same as today's weather. Based on this result we concluded that adaptive schemes capable of adjusting to the current network conditions was the best approach, such control schemes must gamble that current network behaviour is a good indication as to future network behaviour. In Chapter 5 we proposed an adaptive scheduling mechanism based on the use of stochastic learning automata. The scheme was able to adapt such that the performance objectives of the supported flows were just satisfied, thus maximising the resources available for other connections and hence

maximising potential utilisation. The scheme was able to satisfy heterogeneous performance objectives such as mean delays, maximum delays and loss rates and we showed that the automaton was able to converge to the theoretically calculated optimal probabilities. We compared the scheme with a number of alternative algorithms and demonstrated that although sometimes the performance was sub-optimal, it was able to perform well over a much wider range of traffic conditions. In addition we found that the more diverse the traffic conditions were, then the better the scheme performed in comparison to the others. In Chapter 6 we highlighted the fact that because network resources are finite, any scheduling mechanism must be accompanied by a corresponding admission control policy such that the scheduler does not become overloaded. We proposed a mechanism which estimated the required probability for a new flow given the traffic characteristics of the source. For leaky bucket constrained traffic sources the function which maps the traffic characteristics and performance objectives to the required probability is, to the best of our knowledge, unknown and we used a neural network to learn this function based on examples generated empirically. We demonstrated the accuracy of the neural network and compared the overall scheme to static and dynamic WFQ. The static scheme was able to satisfy the performance objectives however it resulted in poor utilisation since the admission control policy is based on conservative assumptions. The dynamic scheme resulted in greater utilisation but a significant number of calls failed to meet their performance objectives. By contrast the proposed scheme was able to achieve satisfactory utilisation and still satisfy the performance objectives.

Overall the main contribution of this thesis is in the application of learning algorithms for providing a measurement based service. A key aspect is that this service is only able to satisfy soft performance guarantees which can accept the occasional performance violation. Such a service will be able to achieve higher utilisation than traditional services which rely on a priori traffic characteristics, although traditional schemes are able to provide much firmer guarantees. It remains an open question as to the popularity of a measurement based service and much depends on the attitudes of the users, since the risk of performance violations will be reflected in a lower cost for the service resulting from the increased utilisation. How conservative any measurement based control scheme will be is also an open question which once again must depend on the attitudes of the users and the price of the service.

7.1 Further Work

In the work presented in this thesis it has been assumed that a logical queue is associated with each traffic flow. In large switches it may be the case that a number of calls with similar QoS requirements will be multiplexed together and the aggregated calls will be treated as a single flow. Such a scenario will complicate the admission control policy since instead of simply allocating a probability to a new flow, it will be necessary to determine how much additional probability should be allocated to a flow consisting of multiplexed calls should a new call request admission. A conservative approach would be to simply allocate the probability that the new call would receive if it was treated independently, however this would be conservative since statistical multiplexing effects would be neglected. An alternative approach would be to combine the flows stochastically based on the mean and variance of the individual flows and base the admission decision on this. This is similar in principle to the H-BIND traffic model described in [36], however this approach does not take advantage of measurements. A final more complicated approach would be to determine the equivalent token bucket characterisation of the existing aggregate flow based on measurements of the aggregate bandwidth usage and experienced delays, this is similar to the approach adopted in [147]. The token bucket parameters of the existing aggregate flow and the new flow are combined to generate an overall worst case characterisation and the admission decision is based on this.

The work presented in this thesis only considers a single node and the obvious extension to the work is the application to network scenarios with the goal being to satisfy end-to-end requirements. Moreover, stationary traffic arrival patterns have generally been assumed and the true effectiveness and robustness of any measurement based scheme can only be determined when applied in a real network scenario in which real sources generate traffic. Furthermore, the non-stationary features of traffic in a real network environment are likely to require that the utilisation factor, which throughout this thesis has been assumed to be unity, will need to be lower in order to meet performance objectives. The optimal value of the utilisation factor can only be determined through experimentation. Also our measurement mechanism is very simple and further work is required to identify the optimal methods in which measurements should be interpreted, for example dividing the measurement interval into smaller sub-intervals and using the maximum observation in these sub-intervals. When applied in a real network it will also be possible to determine the best parameter set for different traffic types, this aspect of tuning parameters has been rather neglected in this thesis since generally the traffic used represents extremes of behaviour and is unlikely to be representative of actual traffic.

Other possible applications of artificial intelligence to the problems of network control include the use of neural networks to learn optimal parameter values in multiplexers given the particular traffic characteristics. As already mentioned, the analysis of some queueing systems becomes very complicated for particular arrival patterns and service disciplines. This results in the functions relating the traffic characteristics and requirements to the required parameters being unknown or requiring a great deal of computational effort. Therefore training a neural network based on simulation results, as with the admission control scheme in Chapter 6, may be beneficial.

Finally genetic algorithms could be applied to some of the optimisation problems associated with network control, for example the problem of selecting packets to serve in an input buffered switch with bypass queueing, this is effectively a maximal matching problem. The notation generally adopted for this problem is one of '0's and '1's and this appears to lend itself to the use of genetic algorithms since they also adopt this notation to encode problems. Therefore genetic algorithms may be able to find a fast near optimal solution in software as opposed the solutions mentioned in Chapter 3 which use neural networks implemented in hardware to tackle the problem. Also genetic algorithms could be used for determining the scheduling order of packets such that some cost function is minimised. An exhaustive search would be far too computationally demanding, however genetic algorithms may be able to reach a near optimal solution in much less time.

Appendix A

Neural Networks - An Overview

In this appendix we provide an introduction to the basic theory of neural networks and the basic architecture we concentrate on is the multi-layer perceptron. More advanced theory on various neural network architectures such as radial basis function networks and Kohonen networks can be found in [83].

A.1 Basic Model of a Neuron

The basic building block of a neural network is a processing element called a neuron (node) (Figure A.1).

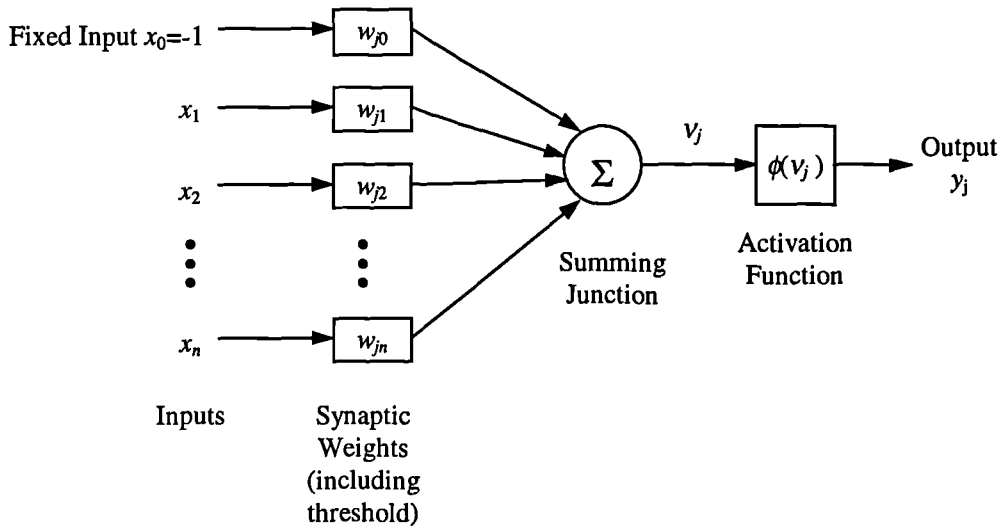

Figure A.1:Model of a typical neuron

Each neuron has one or more inputs and the input signals come from either the external environment or from the outputs of other neurons in the network. Associated with each input connection is an adjustable value called a weight or connection strength. Usually w_{ji} represents the weight of the connection from neuron i to neuron j, and it is these weights that govern the

behaviour of the neural network and various learning algorithms have been developed to adjust the weights such that they can exhibit learning behaviour. Also associated with each neuron is a threshold θ_j and the operation of a single neuron can be described as,

$$y_j = \varphi\left(\sum_{i=1}^{n} x_i w_{ji} - \theta_j\right) \tag{A.1}$$

The function φ in equation (A.1) is called the activation function. The activation function defines the output of a neuron in terms of the activity level and the three basic types of activation functions are threshold functions, piece-wise linear functions and sigmoid functions. The most common activation function is the sigmoid function, which is defined as a strictly increasing function which exhibits smoothness and asymptotic properties. An example of a sigmoid function is the logistic function given in (A.2), where v_j is the net internal activity of neuron j, and y_j is the output of the neuron.

$$y_j = \frac{1}{1+\exp(-v_j)} \tag{A.2}$$

A.2 Standard Multi-Layer Perceptron

A typical feedforward multi-layer perceptron consists of a set of source nodes, known as the input layer, one or more hidden layers of computation nodes, and an output layer. Each neuron is connected to every neuron in the previous layer and the input signals propagate through the network in a forward direction. Such networks have been applied to a variety of problems by training them in a supervised manner using the back-propagation algorithm. This algorithm consists of two phases : a forward pass and a backward pass. In the forward pass, a pattern is applied to the input layer, and its effect propagates through the network until a set of outputs is produced at the output layer. During the backward pass and error signal is generated by calculating the difference between the actual response of the network and the target response. The error signal is then propagated back through the network and the synaptic weights are adjusted so as to make the actual response of the network move closer to the desired response.

The error signal at the output of neuron j is defined by,

$$e_j = d_j - y_j \tag{A.3}$$

where d_j is the desired response and y_j is the actual response. We define the instantaneous value of the squared error for neuron j as,

$$E = \frac{1}{2} e_j^2 \tag{A.4}$$

If the network is trained using a training set of size N, then the total error is given by

$$E_{total} = \sum_{k=1}^{N} E_k \tag{A.5}$$

where E_k denotes the discrepancy of the kth training example. The objective of the training phase is to minimise the E_{total} defined in (A.5). Back-propagation is a gradient descent algorithm, and the weights are adjusted according to an amount proportional to the derivative of E with respect to w.

$$\Delta w_{ji}^k = -\eta \frac{\partial E}{\partial w_{ji}^k} \tag{A.6}$$

where η is the learning parameter. In a strict gradient descent algorithm, the E in (A.6) should be E_{total}, and the updating of the weights should take place after a pass through all of the training set. This is called batch mode back-propagation. However, a more common approach is to update the weights after each training example. A derivation of the back propagation algorithm is given in [83]. It is often the case that the training set needs to be presented to the network several times before the network converges to a stable set of weights. Each presentation of the training set is known as an epoch or an iteration.

A.3 FIR Multi-layer Perceptron

A finite impulse response (FIR) multi-layer perceptron is responsive to time varying signals since it has memory. In this type of neural network each synapse is represented by a finite impulse response filter, this is equivalent to a time-delay neural network in which the outputs of a layer are buffered several time steps and then fed fully connected to the next layer. An FIR multilayer perceptron may be trained by unfolding it in time and performing standard back-propagation, however a more efficient method is to use the temporal back-propagation algorithm that invokes

certain approximations to simplify the computation.

Each FIR filter (synapse) is said to have l taps which represent the delay units. The index l ranges from 0 to M, where M is the total number of delay units built into the filter. According to this model, the signal $s_{ji}(n)$ at the output of the synapse from neuron i to neuron j is given by a linear combination of delayed values of the input signal $x_i(n)$,

$$s_{ji}(n) = \sum_{l=0}^{M} w_{ji}(l) x_i(n-l) \tag{A.7}$$

A synapse with an FIR filter is shown in Figure A.2.

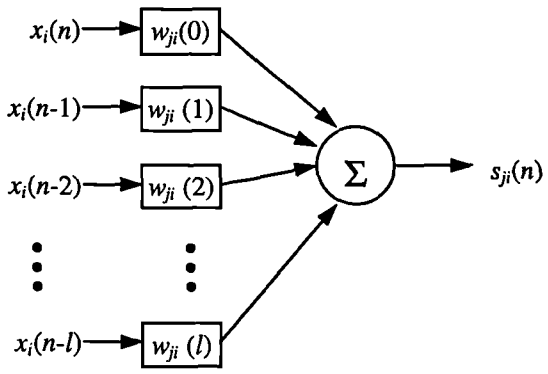

Figure A.2:Synapse from neuron i to neuron j represented as an FIR filter

By introducing the following definitions for the state vector and weight vector for synapse i, respectively, we may thus express the scalar signal $s_{ji}(n)$ as the scalar product of the vectors $w_{ji}(n)$ and $x_i(n)$.

$$\mathbf{x}_i(n) = [x_i(n), x_i(n-1), \ldots\ldots, x_i(n-M)]^T \tag{A.8}$$

$$\mathbf{w}_{ji} = [w_{ji}(0), w_{ji}(1), \ldots\ldots, w_{ji}(M)]^T \tag{A.9}$$

Therefore, summing over the complete set of p synapses, we may describe the output $y_j(n)$ of neuron j by the following equation,

$$y_j(n) = \varphi\left(\sum_{i=1}^{p} s_{ji}(n) - \theta_j\right) = \varphi\left(\sum_{i=1}^{p} \mathbf{w}_{ji}^T \mathbf{x}_i(n) - \theta_j\right) \tag{A.10}$$

Note that the only difference between this and the standard model of the neuron in (A.1) is that the scalars w_{ji} and x_i, are replaced by the weight vector $\boldsymbol{w}_{ji}$ and the state vector $\boldsymbol{x}_i(n)$, respectively. As with the standard model, the aim is to minimise the squared error computed over all time. In a similar way to the standard back-propagation algorithm, the temporal back-propagation algorithm is derived in [83].

Appendix B

Learning Automata - An Overview

A learning automaton can be regarded as a finite state machine. It operates by selecting an action from a finite set of actions which is then evaluated by a random environment. The response from the environment is used by the automaton to select the next action and by this process the automaton learns to select the optimal action. The manner in which the environment response is used to select the next action is determined by the learning algorithm.

A.1 Introduction

An automaton can be described mathematically by a quintuple,

$$A \equiv \{\alpha, \beta, \mathbf{F}, \mathbf{G}, \phi\} \tag{B.1}$$

The vector α forms the output or action set of the automaton, the vector β defines the inputs, the vector ϕ defines the internal state, the transition function **F** determines the state at instant n+1 given the inputs and state at instant n and the transition function **G** determines the output of the automaton given the current state.

We are particularly interested in stochastic automata in which the functions **F** and **G** are stochastic and probabilities are associated with the next states and outputs of the automaton. In this type of automaton the internal state of the automaton, ϕ, is replaced by the action probability vector, **p**, which is defined as,

$$\mathbf{p}(n) = \{p_1(n), p_2(n), p_3(n), \ldots\ldots, p_r(n)\} \tag{B.1}$$

The particular automaton described in (B.1) has r actions and,

$$p_i(n) = \text{Prob}[\alpha(n) = \alpha_i] \tag{B.2}$$

where,

$$\sum_{i=1}^{r} p_i(n) = 1 \tag{B.3}$$

The environment can be mathematically described by the triple,

$$E \equiv \{\alpha, \beta, \mathbf{c}\} \tag{B.4}$$

where α is the set of inputs to the environment, β is the set of outputs and $\mathbf{c}$ is the set of penalty probabilities. The input to the environment is one of the r actions selected by the automaton. If the action is i then the output of the environment is given by β_i and the environment can be categorised according to the type of response it generates. A P-Model type generates a binary response, in such an environment β_i=1 is taken as a failure while β_i=0 is taken as a success. In the Q-Model environment, β_i(n) can take on a finite number of values in the range [0,1]. An S-Model environment can generate any variable in the range [0,1] i.e. $\beta_i(n) \in [0,1]$. The set of penalty parameters, $\mathbf{c}$, characterises the environment and is defined as,

$$c_i = \text{Prob}[\beta(n) = 1 \mid \alpha(n) = \alpha_i] \tag{B.5}$$

This is the probability that action α_i would result in an unfavourable response from the environment and the values of c_i are unknown. It is the aim of the automaton to minimise each c_i such that the probability of a failure is also minimised.

The connection of the stochastic automaton and the environment is shown in Figure B.1 and together with a learning algorithm forms a stochastic learning automaton.

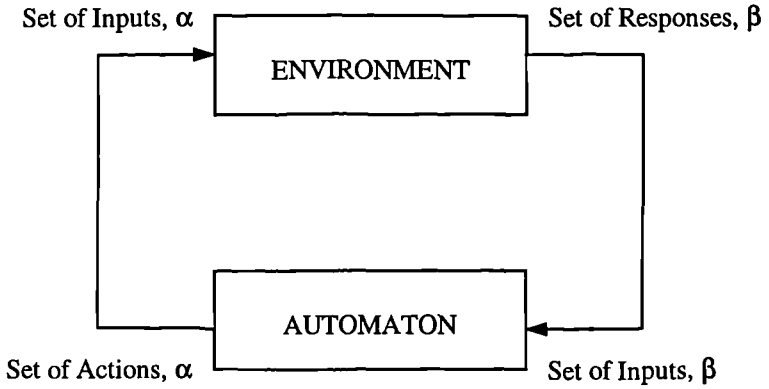

Figure B.1:Learning Automaton

A stochastic learning automaton can be described by the quintuple,

$$SLA \equiv \{\alpha, \beta, \mathbf{p}, \mathbf{T}, \mathbf{c}\} \tag{B.6}$$

where α is the set of outputs of the automaton / inputs to the environment, β is the set of inputs to the automaton / outputs from the environment, $\mathbf{p}$ is the probability vector, $\mathbf{T}$ is the learning algorithm and $\mathbf{c}$ is the set of penalty probabilities which define the environment.

A.2 Performance Measures

A stationary environment can be defined as one in which the set of penalty probabilities, $\mathbf{c}$, remains constant over time. By contrast, a non-stationary environment is characterised by a penalty set that varies over time. A special case of a non-stationary environment is the non-autonomous environment where the actions chosen influence the values of the penalty set. This is the case for the scheduler described in the thesis since selecting a particular queue for service may decrease the attractiveness of selecting that queue in the near future.

To quantify the performance of stochastic learning automata, certain measures have been defined which determine the effectiveness of the automaton and enables the comparison with other schemes. A quantity $M(n)$ is defined as the average penalty received by the automaton for a given action probability vector and is given by,

$$M(n) = E[\beta(n) \mid \mathbf{p}(n)] = \sum_{i=1}^{r} c_i p_i(n) \tag{B.7}$$

An pure-chance automaton that selects each action with equal probability has an average penalty, M_0, given by,

$$M_0 = \frac{1}{r}\sum_{i=1}^{r} c_i \tag{B.8}$$

An automaton which is said to learn must perform better than a pure-chance automaton and the average penalty must be less than M_0. Since $M(n)$ is a random variable, the expected value of $M(n)$ must be compared with M_0 and based on the comparison we can define three types of behaviour.

An automaton is said to be expedient if,

$$\lim_{n\to\infty} E[M(n)] < M_0 \tag{B.9}$$

Therefore a learning automaton is expedient if it performs better than a pure chance automaton. A learning automaton is said to be optimal if,

$$\lim_{n\to\infty} E[M(n)] = c_l \tag{B.10}$$

where $c_l = \min_i\{c_i\}$. Optimality is a desirable feature in a stationary environment since it ensures that the optimal action is chosen with probability 1. However, in a non-stationary environment in which the optimal action may change an ε-optimal automaton is more desirable since it ensures that the automaton does not get locked into any particular state. An ε-optimal automaton chooses the optimal action with a probability arbitrarily close to 1 depending on the choice of ε.

$$\lim_{n\to\infty} E[M(n)] = c_l + \varepsilon \tag{B.11}$$

B.3 Learning Algorithms

The learning algorithm determines how the action probability is modified with respect to the performed action, $\alpha(n)$, and the environment response, $\beta(n)$. The learning algorithm can be represented as,

$$\mathbf{p}(n+1) = \mathbf{T}[\mathbf{p}(n), \alpha(n), \beta(n)] \tag{B.12}$$

If the function **T** is linear then the learning algorithm is said to be linear, otherwise it is said to be non-linear. The idea behind any learning algorithm is that if the automaton selects a particular action and obtains a favourable response from the environment then the corresponding action probability is increased while the action probabilities of the other actions are decreased. Similarly, for an unfavourable response the corresponding action probability is decreased while the others are increased.

If $\alpha(n)= \alpha_i$, then a general linear algorithm for a P-Model environment can be defined as follows,

If $\beta(n)=0$ (favourable),

$$p_i(n+1) = p_i(n) + a[1 - p_i(n)] \tag{B.13}$$

$$p_j(n+1) = (1-a)p_j(n) \qquad \forall j\,;\, j \neq i \tag{B.14}$$

If $\beta(n)=1$ (unfavourable),

$$p_i(n+1) = (1-b)p_i(n) \tag{B.15}$$

$$p_j(n+1) = \frac{b}{1-r} + (1-b)p_j(n) \qquad \forall j\,;\, j \neq i \tag{B.16}$$

where a and b are the reward and penalty parameters respectively. If $a=b$ then the updating is known as the Linear Reward Penalty (L_{RP}) scheme. If $b=0$, the Linear Reward Inaction (L_{RI}) scheme is obtained. Finally if $b<<a$ then the resulting scheme is known as the Linear Reward Epsilon Penalty ($L_{R\varepsilon P}$) scheme.

Other learning algorithms include discretised algorithms in which the action probabilities are limited to taking on discrete values in range [0,1]. The idea behind such algorithms is that they approach the optimum directly rather than asymptotically as with continuous algorithms, the trade off between the rate of convergence and the stability upon convergence can be controlled by the degree of quantisation. Estimator algorithms maintain an estimate of the penalty probabilities as the learning proceeds. The internal state of the automaton is now characterised by the estimate of the penalty probabilities in addition to the action probabilities, and this additional information is used in the updating of the probabilities and generally yields a faster rate of convergence at the expense of additional storage.

Appendix C

MPEG-I Coding

Due to the high bandwidth needs of uncompressed video streams, several coding algorithms for the compression of these streams have been developed. The MPEG-I coding algorithm achieves compression by reducing both the temporal and spatial redundancy of the video stream. The spatial redundancies are reduced by entropy coding and transforms, and the temporal redundancies are reduced by the prediction of future frames based on motion vectors. This prediction is achieved by using three types of frames.

- **I-frames** use only intra-frame coding, based on discrete cosine transform and entropy coding. This frame is coded without reference to other frames, therefore allowing random access.
- **P-frames** are similar to I-frames, but with the addition of motion compensation with respect to the previous I- or P- frame.
- **B-frames** are similar to P-frames, except that the motion compensation can be with respect to the previous I- or P- frame, the next I- or P- frame, or an interpolation between the two.

I-frames are generally larger than P-frames, which in turn are generally larger than B-frames. After coding, the frames are arranged in the periodic sequence "IBBPBBPBBPBB", this is called a *group of pictures* (GOP). A summary of MPEG-I coding is given in Figure 10.1.

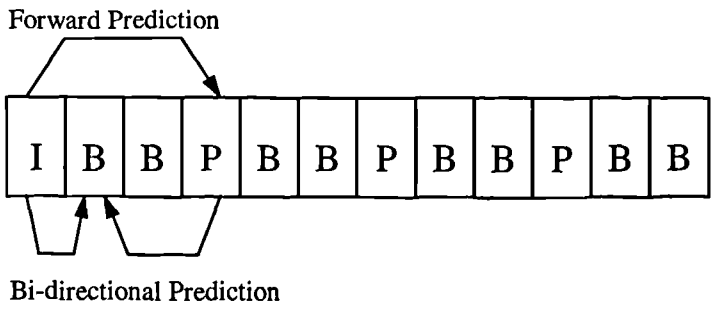

Figure 10.1:Group of pictures in an MPEG-I stream

Appendix D

Discrete Time M/M/1 and MMDP/M/1 Queues

In this appendix we derive solutions to discrete time queueing systems with geometric service times and arrivals that follow Bernoulli and Markov Modulated Deterministic Processes (MMDP).

D.1 Bernoulli Arrival Process

Consider the case when the arrival process at the queue is Bernoulli. The average arrival rate is given by λ, therefore the probability that a cell arrives in any given time slot is also given by λ. The average service rate is μ, therefore the probability that a cell is served in a given time slot is also given by μ. We shall define,

$q(n)$ - Probability that there are n cells in the queue at any given time

$d(n)$ - Probability that a cell experiences a delay of n time slots passing through the queue

The Markov chain for this queuing process is given in Figure 11.1.

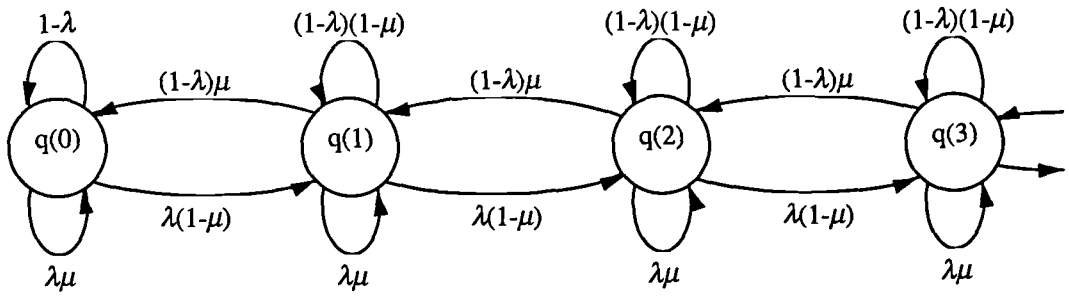

Figure 11.1:Markov chain for queuing process

The flow balance equation for this chain is,

$$q(n)\lambda(1-\mu) = q(n+1)(1-\lambda)\mu \tag{D.1}$$

Thus,

$$q(n+1) = \frac{\lambda(1-\mu)}{(1-\lambda)\mu} q(n) = \rho q(n) \qquad \text{where } \rho = \frac{\lambda(1-\mu)}{(1-\lambda)\mu} \tag{D.2}$$

Repeating equation (D.2) n times, one finds that,

$$q(n) = \rho^n q(0) \tag{D.3}$$

To find $q(0)$, we must invoke the probability normalisation condition,

$$\sum_n q(n) = 1 \tag{D.4}$$

For an infinite queue, one finds that if $\rho<1$,

$$q(0) = 1-\rho \tag{D.5}$$

therefore,

$$q(n) = (1-\rho)\rho^n \tag{D.6}$$

Note that since the probability that the queue is empty is $q(0)=1-\rho$, the probability that the queue is non-empty is just the utilisation ρ. This result is the same as with an M/M/1 queue but with a different ρ, therefore it is possible to calculate various statistics of the queue length distribution using the M/M/1 queuing equations.

However, we are interested in the delay distribution. For the system we have, an arriving cell will suffer zero delay only if the queue is empty and the server serves the arriving cell immediately, therefore,

$$d(0) = q(0)\mu \tag{D.7}$$

Similarly, if the cell suffers a delay of one time slot, one of two situations must arise. Either the cell arrives at a queue containing one cell and the both cells are served consecutively, or the cell arrives at an empty queue and must wait one time slot before being served.

$$d(1) = q(0)(1-\mu)\mu + q(1)\mu^2 \tag{D.8}$$

Similarly,

$$d(2) = q(0)(1-\mu)^2\mu + 2q(1)(1-\mu)\mu^2 + q(2)\mu^3 \tag{D.9}$$

This gives us the general result,

$$d(n) = \sum_i q(i)\mu\binom{n}{i}\mu^i(1-\mu)^{n-i} \tag{D.10}$$

Further manipulation yields,

$$d(n) = k(1-k)^n \qquad \text{where } k = \frac{\mu-\lambda}{1-\lambda} \tag{D.11}$$

Thus, the mean delay is given by,

$$\bar{d} = \frac{1-k}{k} = \frac{1-\mu}{\mu-\lambda} \tag{D.12}$$

Therefore given a mean delay requirement of $\bar{d}$, the required service rate is given by,

$$\mu = \frac{1+\bar{d}\lambda}{1+\bar{d}} \tag{D.13}$$

Also since the delay distribution is geometric then the cumulative delay distribution is given by,

$$D(n) = 1-(1-k)^{n+1} \tag{D.14}$$

Therefore it is possible to calculate the pth percentile of the distribution. Using this we can calculate the required service rate such that p% of the cells suffer a delay less than n, this is given by,

$$\mu = \left(1-(1-p)^{1/n+1}\right)(1-\lambda)+\lambda \tag{D.15}$$

D.2 Markov Modulated Deterministic Process

An MMDP is the most common bursty traffic model with the generating process alternating between two states. The OFF state corresponds to no cell arrivals. The ON state corresponds to

cells arriving in every time slot. The process is described by two parameters, α and β, that represent the probabilities of leaving the ON and OFF states, respectively. Furthermore the mean ON and OFF periods are given by $1/\alpha$ and $1/\beta$, respectively. The average arrival rate, λ, defined as the probability that a cell arrives in any given time slot, is related to α and β as follows,

$$\lambda = \frac{\beta}{\alpha + \beta} \tag{D.16}$$

The model is summarised in Figure 11.2.

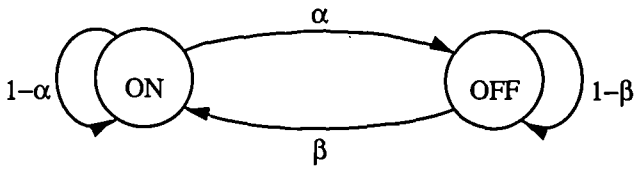

Figure 11.2 : Markov chain describing traffic arrival process

In contrast to the Bernoulli arrival process, the overall Markov process is two-dimensional in order to take into account the ON and OFF states (Figure 11.3). Define the following,

$q(n,0)$ - Probability that the queue occupancy is n cells and the source is in the OFF state

$q(n,1)$ - Probability that the queue occupancy is n cells and the source is in the ON state

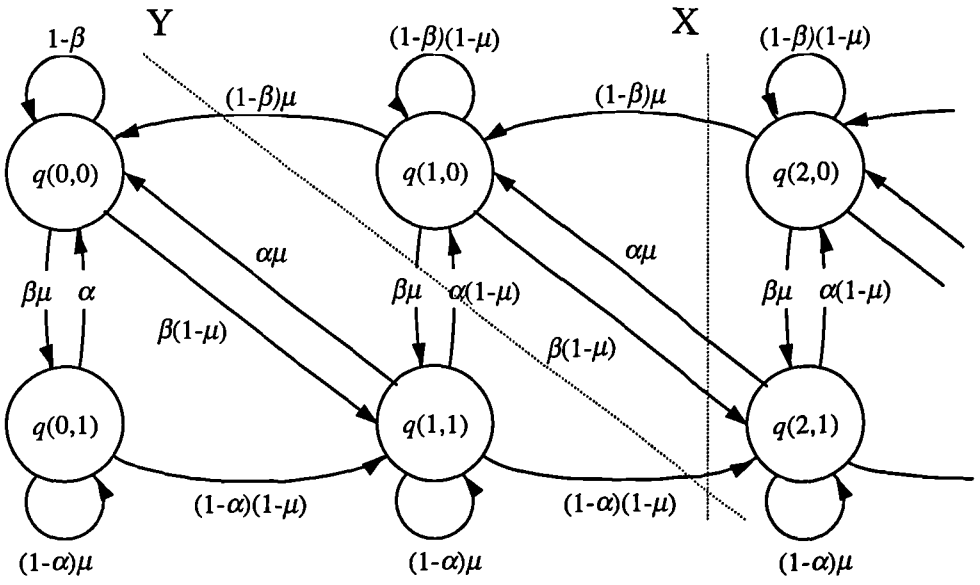

Figure 11.3 : Markov chain for queuing process with MMDP arrivals

The flow balance equations are generated by considering the probability flows across the planes labelled X and Y. Considering plane X we obtain,

$$q(n+1,0)(1-\beta)\mu + q(n+1,1)\alpha\mu = q(n,0)\beta(1-\mu) + q(n,1)(1-\alpha)(1-\mu) \tag{D.17}$$

Similarly with plane Y we obtain,

$$q(n,0)(1-\beta)\mu + q(n,0)\beta\mu = q(n,1)\alpha(1-\mu) + q(n,1)(1-\alpha)(1-\mu) \tag{D.18}$$

Further manipulation of (D.18) yields,

$$q(n,0) = q(n,1)\frac{(1-\mu)}{\mu} \tag{D.19}$$

We can define,

$$q(n,0) + q(n,1) = q(n) \tag{D.20}$$

Substituting (D.19) into (D.20) gives us,

$$q(n,1) = q(n)\mu \tag{D.21}$$

$$q(n,0) = q(n)(1-\mu) \tag{D.22}$$

Substituting (D.21) and (D.22) into (D.17) gives us the result,

$$q(n+1) = \frac{\mu(1-\mu)(1-\alpha) + (1-\mu)^2\beta}{\mu(1-\mu)(1-\beta) + \mu^2\alpha} q(n) \tag{D.23}$$

As before we obtain the result,

$$q(n) = (1-\rho)\rho^n \tag{D.24}$$

where,

$$\rho = \frac{\mu(1-\mu)(1-\alpha) + (1-\mu)^2\beta}{\mu(1-\mu)(1-\beta) + \mu^2\alpha} \tag{D.25}$$

The delay distribution is obtained using (D.11) and is given by,

$$d(n) = k(1-k)^n \qquad \text{where } k = \mu(1-\rho) \tag{D.26}$$

Using this it is possible to calculate the required service rate such that a particular mean delay requirement is met, this is given by,

$$\mu = \frac{\beta\bar{d} + 1}{\alpha\bar{d} + \beta\bar{d} + 1} \tag{D.27}$$

Also the required service rate in order to ensure that $p\%$ of the cells suffer a delay less than n is given by,

$$\mu = \frac{1 - X + \beta X}{1 - X + \beta X + \alpha X} \qquad \text{where } X = 1 - (1-p)^{1/n+1} \tag{D.28}$$

Appendix E

Publications

[1] J. Hall, P. Mars, "A critical review of buffer allocation policies in an ATM switch", *Proceedings of the Fourteenth IEE UK Teletraffic Symposium*, pp. 23/1-23/6, Manchester, 1997.

[2] J. Hall, P. Mars, "Adaptive scheduling to satisfy quality of service using learning algorithms", *Proceedings of the Fifteenth IEE UK Teletraffic Symposium,* Durham, 1998.

[3] J. Hall, P. Mars, "Satisfying QoS with a learning based scheduling algorithm", *Proceedings of the Sixth IEEE/IFIP International Workshop on Quality of Service*, pp. 171-173, Napa, USA, 1998.

[4] J. Hall, P. Mars, "The limitations of artificial neural networks for traffic prediction", *Proceedings of the Third IEEE Symposium on Computers and Communications*, pp. 8-12, Athens, Greece, 1998.

[5] J. Hall, P. Mars, "Satisfying Quality of Service with a learning based multiplexer scheduling algorithm in integrated services packet networks", submitted to *IEE Proceedings Communications*, August 1998.

[6] J. Hall, P. Mars, "The limitations of artificial neural networks for traffic prediction in broadband networks", submitted to *IEE Proceedings Communications*, August 1998.

References

[1] R. Jain, "Congestion control in computer networks : Issues and trends", *IEEE Network*, pp. 24-30, May 1990.

[2] S. Shenker, "Fundamental design issues for the future Internet", *IEEE Journal on Selected Areas in Communications*, Vol. 13, No. 7, pp. 1176-1188, 1995.

[3] M. W. Garrett, "A service architecture for ATM : From applications to scheduling", *IEEE Network*, pp. 6-14, May/June 1996.

[4] D. D. Clark, S. Shenker, L. Zhang, "Supporting real-time applications in an integrated services packet network : Architecture and mechanism", *Proceedings of ACM SIGCOMM '92*, pp. 14-26, 1992.

[5] S. Shenker, D. D. Clark, L. Zhang, "A scheduling service model and scheduling architecture for an integrated services packet network",

[6] D. Ferrari, D. Verma, "A scheme for real-time channel establishment in wide-area networks", *IEEE Journal on Selected Areas in Communications*, Vol. 8, No. 3, pp. 368-379, 1990.

[7] R. Nagarajan, J. F. Kurose, "On defining, computing, and guaranteeing quality-of-service in high-speed networks", *Proceedings of IEEE Infocom '92*, pp. 2016-2025, 1992.

[8] S. Shenker, C. Partridge, R. Guerin, "Specification of guaranteed quality of service", RFC-2212, 1997.

[9] J. Wroclawski, "Specification of the controlled-load network element service", RFC-2211, 1997.

[10] C. A. Cooper, K. I. Park, "Towards a broadband congestion control strategy", *IEEE Network*, pp. 18-23, May 1990.

[11] R. O. Onvural, *Asynchronous Transfer Mode Networks*, Artech House, 1994.

[12] D. E. Wrege, E. W. Knightly, H. Zhang, J. Liebeherr, "Deterministic delay bounds for VBR video in packet-switching networks : Fundamental limits and practical tradeoffs", *IEEE/ACM Transactions on Networking,* Vol. 4, No. 3, pp. 352-362, 1996.

[13] R. Cruz, "A calculus for network delay, Part I : Network elements in isolation", *IEEE Transactions on Information Theory*, Vol. 37, No. 1, pp. 114-121, 1991.

[14] H. Heffes, D. M. Lucantoni, "A Markov modulated characterisation of packetised voice

and data traffic and related statistical multiplexer performance", *IEEE Journal on Selected Areas in Communications,* Vol. 4, No. 6, pp. 856-868, 1986.

[15] K. Sriram, W. Whitt, "Characterising superposition arrival processes in Packet Multiplexers for voice and data", *IEEE Journal on Selected Areas in Communications*, Vol. 4, No. 6, pp. 833-846, 1986.

[16] W-C. Lau, A. Erramilli, J. Wang, W. Willinger, "Self-similar traffic generation: The random midpoint displacement algorithm and it's properties", *Proceedings of ICC*, pp. 466-472, June 1995.

[17] V. Paxson, "Fast approximation of self-similar network traffic", Technical Report, Lawrence Berkeley Laboratory, 1995 (*http://www-nrg.ee.lbl.gov/nrg-papers.html*).

[18] ATM Forum Traffic Management Specification Version 4.0, 1996.

[19] S. Low, P. Varaiya, "A simple theory of traffic and resource allocation in ATM", *Proceedings of IEEE Globecom '91*, pp. 1633-1637, 1991.

[20] E. W. Knightly, H. Zhang, "D-BIND : An accurate traffic model for providing QoS guarantees to VBR traffic", *IEE/ACM Transactions on Networking*, Vol. 5, No. 2, pp. 219-231, 1997.

[21] V. S. Frost, B. Melamed, "Traffic modelling for telecommunications networks", *IEEE Communications Magazine*, pp. 70-81, March 1994.

[22] G. D. Stamoulis, M. E. Anagnostou, A. D. Georganta, "Traffic source models for ATM networks : A survey", *Computer Communications*, Vol. 17, No. 6, pp. 428-438, 1994.

[23] B. Maglaris, D. Anastassiou, P. Sen, G. Karlsson, J. D. Robbins, "Performance models of statistical multiplexing in packet video communications", *IEEE Transaction on Communications*, Vol. 36, No. 7, pp. 834-843, 1988.

[24] P. Sen, B. Maglaris, N-E, Rikli, D. Anastassiou, "Models for packet switching of variable-bit-rate video sources", *IEEE Journal on Selected Areas in Communications*, Vol. 7, No. 5, pp. 865-869, 1989.

[25] D. P. Heyman, A. Tabatabai, T. V. Lakshman, "Statistical analysis and simulation study of video teleconference traffic in ATM networks", *IEEE Transactions on Circuits and Systems for Video Technology*m Vol. 2, No. 1, pp. 49-58, 1992.

[26] D. P. Heyman, T. V. Lakshman, "Source models for VBR broadcast video traffic", *IEEE/ACM Transactions on. Networking*, Vol. 4, No. 1, pp. 40-48, 1996.

[27] W. Leland, M. Taqqu, W. Willinger and D. Wilson, "On the self-similar nature of Ethernet traffic (Extended version)", *IEEE/ACM Transactions on Networking*, Vol. 2, No. 1, pp. 1-15, 1994.

[28] J. Beran, R. Sherman, M. S. Taqqu, W. Willinger, "Long-range dependence in variable-bit-rate video traffic", *IEEE Transactions on Communications*, Vol. 43, No. 2/3/4, pp. 1566-1579, 1995.

[29] V. Paxson, S. Floyd, "Wide area traffic : The failure of Poisson modelling", *IEEE/ACM Transactions on Networking*, Vol. 3, No. 3, pp. 226-244, 1995.

[30] D. R. Cox, "Long-range dependence : A review", In *Statistics : An appraisal*, pp. 55-74, The Iowa State Univeristy Press, 1984.

[31] I. Norros, "A storage model with self-similar input", *Queueing Syatems Theory and Applications,* Vol. 16, pp. 387-396, 1994.

[32] Internet Traffic Archive WWW site, *http://ita.ee.lbl.gov/index.html*

[33] J. Kurose, "On computing per-session performance bounds in high-speed multi-hop computer networks", *Proceedings of ACM SIGMETRICS '92*, pp. 128-139, 1992.

[34] H. Zhang, E. W. Knightly, "Providing end-to-end statistical performance guarantees with bounding interval dependent stochastic models", *Proceedings of ACM SIGMETRICS '94*, pp. 211-220, 1994.

[35] O. Yaron, M. Sidi, "Performance and stability of communication networks via robust exponential bounds", *IEEE Transactions on Networking*, Vol. 1, No. 3, pp. 372-385, 1993.

[36] E. W. Knightly, "H-BIND : A new approach to providing statistical performance guarantees to VBR traffic", *Proceedings of IEEE Infocom '96,* pp. 1091-1099, 1996.

[37] H. Zhang, D. Ferrari, "Rate-controlled service disciplines", *Journal of High Speed Networks*, Vol. 3, No. 4, 1994.

[38] J. M. Hyman, A. A. Lazar, G. Pacifici, "A separation principle between scheduling and admission control for broadband switching", *IEEE Journal on Selected Areas in Communications*, Vol. 11, No. 5, pp. 605-616, 1993.

[39] A. Demers, S. Keshav, S. Shenker, "Analysis and simulation of a fair-queueing algorithm", *Proceedings of ACM SIGCOMM '89*, pp. 1-12, 1989.

[40] H. Zhang, "Service disciplines for guaranteed performance service in packet switching networks", *Proceedings of the IEEE*, Vol. 83, No. 10, pp. 1374-1396, 1995.

[41] D-S. Lee, B. Sengupta, "Queueing analysis of a threshold based priority scheme for ATM networks", *IEEE/ACM Transactions on Networking,* Vol. 1, No. 6, pp. 709-717, 1993.

[42] S. A. Banawan, G. Radhakrishnan, "An adaptive threshold based scheduling policy for ATM networks", *Conference Proceedings of the 1996 IEEE 15th Annual International Pheonix Conference on Computers and Communications,* Ch. 76, pp. 439-445, 1996.

[43] D-S. Lee, "Generalised longest queue first : An adaptive schedulinh discipline for ATM networks", *Proceedings of IEEE INFOCOM '97*, pp. , 1997.

[44] C. Lui, J. Layland, "Scheduling algorithms for multiprogramming in a hard real time environment", *Journal of ACM*, Vol. 20, No.1, pp. 46-61, 1973.

[45] L. Georgiadid, R. Guerin, A. Parekh, "Optimal multiplexing on a single link : Delay and buffer requirements", *IEEE Transactions on Information Theory*, Vol. 43, No. 5, 1997.

[46] Y. Lim, J. E. Kobza, "Analysis of a delay dependent priority discipline in an integrated multiclass traffic fast packet switch", *IEEE Transactions on Communications,* Vol. 38, No. 5, pp. 659-665, 1990.

[47] J. Liebeherr, D. E. Wrege, D. Ferrari, "Exact admission control for networks with a bounded delay service", *IEEE/ACM Transactions on Networking,* Vol. 4, No. 6, pp. 885-901, 1996.

[48] T-Y. Huang, J-L. C. Wu, "Performance analysis of a dynamic priority scheduling method in ATM networks", *IEE Proceedings*, Vol. 140, No. 4, pp. 285-290, 1993.

[49] L. P. Clare, A. R. K. Sastry, "Value-based multiplexing of time critical traffic", *Proceedings of IEEE MILCOM '89,* pp. 395-401, 1989.

[50] J. M. Peha, F. A. Tobagi, "Cost-based scheduling and dropping algorithms to support integrated services", *IEEE Transactions on Communications,* Vol. 44, No. 2, pp. 192-201, 1996.

[51] J. M. Peha, "Heterogeneous-criteria scheduling : Minimising weighted number of tardy jobs and weighted completion time", *Journal of Computers and Operations Research,* Vol. 22, No. 10, pp. 1089-1100, 1995.

[52] M. Katevenis, S. Sidiropoulos, C. Courcoubetis, "Weighted round-robin cell multiplexing in a general-purpose ATM switch chip", *IEEE Journal on Selected Areas in Communications,* Vol. 9, No. 8, pp. 1265-1279, 1991.

[53] L. Zhang, "Virtual Clock : A new traffic control algorithm for packet switching networks", *ACM Transactions on Computer Systems*, Vol. 9, No. 2, pp. 101-124, 1991.

[54] A. K. Parekh, R. G. Gallager, "A generalized processor sharing approach to flow control in integrated services networks: The single node case", *IEEE/ACM Transactions on Networking*, Vol. 1, No. 3, pp. 344-357, 1993.

[55] J. C. R. Bennett, H. Zhang, "WF^2Q : Worst-case fair weighted fair queueing", *Proceedings of IEEE INFOCOM*, pp.120-128, 1996.

[56] S. Golestani, "A self-clocked fair queueing scheme for broadband applications", *Proceedings of IEEE INFOCOM*, pp. 636-646, 1994.

[57] D. Verma, H. Zhang, D. Ferrari, "Guaranteeing delay jitter bounds in packet switching networks", *Proceedings of Tricom*, pp 35-46, 1991.

[58] S. Golestani, "A stop-and-go queueing framework for congestion management", *Proceedings of ACM SIGCOMM*, pp 8-18, 1990.

[59] C. Kalmanek, H. Kanakia, S. Keshav, "Rate controlled servers for very high speed networks", *Proceedings of IEEE GLOBECOM*, pp. 12-20, 1990.

[60] H. Zhang, D. Ferrari, "Rate-controlled static priority queueing", *Proceedings of IEEE INFOCOM*, pp. 227-236, 1993.

[61] H. Kroner, G. Hebuterne, P. Boyer, A. Gravey, "Priority management in ATM switching nodes", *IEEE Journal on Selected Areas in Communications*, Vol. 9, No. 3, pp. 418-427, 1991.

[62] H. Zhang, D. Ferrari, "Improving utilisation for deterministic service in multimedia communication", *1994 IEEE International Conference on Multimedia Computing and Systems*, pp. 295-304, 1994.

[63] R. Guerin, H. Ahmadi, M. Naghshineh, "Equivalent capacity and its application to bandwidth allocation in high speed networks", *IEEE Journal on Selected Areas in Communications*, Vol. 9, No. 7, pp. 968-981, 1991.

[64] D. Anick, D. Mitra, M. M. Sondhi, "Stochastic theory of a data-handling system with multiple sources", *Bell System Technical Journal*, Vol. 61, No. 8, pp. 1871-1894, 1982.

[65] A. Elwalid, D. Mitra, "Effective bandwidth for general markovian traffic sources and admission control of high speed networks", *IEEE/ACM Transactions on Networking*, Vol. 1, No. 3, pp. 329-343, 1993.

[66] Z. Fan, P. Mars, "Accurate approximation of cell loss probability for self-similar traffic in ATM networks", *Electronics Letters*, Vol. 32, No. 19, pp. 1749-1751, 1996.

[67] G. Gallassi, G. Rigolio, L. Frata, "ATM : Bandwidth assignment and bandwidth enforcement policies", *Proceedings of IEEE Globecom*, pp. 1788-1793, 1989.

[68] K. Dubose, H. S. Kim, "An effective bit rate / table lookup based admission control strategy for the ATM B-ISDN", *Proceedings of LCN '92*, pp. 20-29, 1992.

[69] T. Murase, H. Suzuki, S. Sato, T. Takeuchi, "A call admission control for ATM networks using a simple quality estimate", *IEEE Journal on Selected Areas in Communications*, Vol. 9, No. 9, pp. 1461-1470, 1991.

[70] T-H. Lee, K-C. Lai, S-T. Duann, "Design of a real-time call admission controller for ATM networks", *IEEE/ACM Transactions on Networking*, Vol. 4, No. 5, pp. 758-765, 1996.

[71] J. W. Causey, H. S. Kim, "Comparison of admission control schemes in ATM networks", *International Journal of Communication Systems*, Vol. 8, pp. 165-184, 1995.

[72] Z-L. Zhang, D. Towsley, J. Kurose, "Statistical analysis of Generalised Processor Sharing Scheduling Discipline", *IEEE Journal on Selected Areas in Communications*, Vol. 13, No. 6, pp. 1071-1080, 1995.

[73] O. Yaron, M. Sidi, "Generalised processor sharing networks with exponentially bounded burstiness arrivals", *Journal of High Speed Networks*, Vol. 13, pp. 375-387, 1994.

[74] E. W. Knightly, "Second moment resource allocation in multi-service networks", *Proceedings of ACM SIGMETRICS '97*, pp. 181-191, 1997.

[75] J. M. Hyman, A. A. Lazar, G. Pacifici, "Real-time scheduling with quality of service constraints", *IEEE Journal on Selected Areas in Communications*, Vol. 9, No. 7, pp. 1052-1063, 1991.

[76] H. Saito, K. Shiomoto, "Dynamic call admission control in ATM networks", *IEEE Journal on Selected Areas in Communications*, Vol. 9, No. 7, pp. 982-989, 1991.

[77] H. Zhang, E. W. Knightly, "RED-VBR : A re-negotiation based approach to support delay-sensitive VBR video", *ACM Multimedia Systems Journal*, Vol. 5, No. 3, pp. 164-176, 1997.

[78] K. Lee, "Observation based multiplexing of VBR traffic in ATM networks", *Proceedings of IEEE Globecom '96*, pp. 1236-1242, 1996.

[79] H. G. Perros, K. M. Elsayed, "Call Admission Control Schemes : A review", *IEEE Communications Magazine*, pp. 82-91, November, 1996.

[80] Z. Haas, J. H. Winters, "Congestion control by adaptive admission", *Proceedings of IEEE Infocom '91*, pp. 560-569, 1991.

[81] S. Russell, P. Norvig, *Artificial Intelligence : A modern approach*, Prentice Hall, 1995.

[82] G. F. Luger, W. A. Stubblefield, *Artificial Intelligence : Structures and Strategies for Complex Problem Solving*, Addison Wesley, 1998.

[83] S. Haykin, *Neural Networks: A Comprehensive Foundation*, Macmillan College Publishing Company, 1994.

[84] D. Hammerstrom, "Neural networks at work", *IEEE Spectrum Magazine*, pp. 26-32, June 1993.

[85] A. Hiramatsu, "Integrated ATM traffic control by distributed neural networks", *Proceedings of the International Switching Symposium '90*, pp. 54-65, 1990.

[86] A. Hiramatsu, "Training techniques for neural network applications in ATM", *IEEE Communications Magazine*, pp. 58-67, October 1995.

[87] Z. Fan, P. Mars, "Application of artificial neural networks to effective bandwidth estimation in ATM networks", *Proceedings of IEEE International Conference on Neural Networks '96*, pp. 1951-1956, 1996.

[88] A. Hiramatsu, "ATM communications network control by neural networks", *IEEE Transactions on Neural Networks,* Vol. 1, No. 1, pp. 122-130, 1990.

[89] R. J. T. Morris, B. Samadi, "Neural network control of communication systems", *IEEE Transactions on Neural Networks,* Vol. 5, No. 4, pp. 639-650, 1994.

[90] A. Hiramatsu, "ATM call admission control using a neural network trained with the output buffer method", *Proceedings of IEEE International Conference on Neural Networks '94*, pp. 3611-3616, 1994.

[91] E. Nordstrom, "A hybrid admission control scheme for broadband ATM traffic", *Proceedings of the International Workshop on the Applications of Neural Networks to Telecommunications*, pp. 77-84, 1993.

[92] C. J. Chang, S. Y. Lin, R. G. Cheng, "PSD-based neral-net connection admission control", *Proceedings of IEEE Infocom '97*, pp. 955-962, 1997.

[93] O. Gallmo, L. Asplund, "Reinforcement learning by construction of hypothetical targets", *Proceedings of the International Workshop on Applications of Neural Networks to Telecommunications 2*, pp. 300-307, 1995.

[94] E. Nordstrom, J. Carlstrom, "A reinforcement learning scheme for adaptive link allocation in ATM networks", *Proceedings of the International Workshop on Applications of Neural Networks to Telecommunications 2*, pp. 88-95, 1995.

[95] A. Hiramatsu, "Integration of ATM call admission control and link capacity control by distributed neural networks", *IEEE Journal on Selected Areas in Communications*, Vol. 9, no. 7, pp. 1131-1138.

[96] A. Tarraf, I. Habib, T. Saadawi, "A novel neural network traffic enforcement mechanism for ATM networks", *IEEE Journal on Selected Areas in Commincations*, Vol. 12, No. 6, pp. 1088-1096, 1994.

[97] A. D. Estrella, E. Casilari, A. Jurado, F. Sandoval, "ATM traffic neural control : Multiservice call admission and policing function", *Proceedings of the International Workshop on Applications of Neural Networks to Telecommunications 2*, pp. 104-111, 1995.

[98] I. Habib, A. Tarraf, T. Saadawi, "A neural network controller for congestion control in ATM multiplexers", *Computer Networks and ISDN Systems,* Vol. 29, pp. 325-334, 1997.

[99] Z. Fan, P. Mars, "Access flow control scheme for ATM networks using neural-network-

based traffic prediction", *IEE Proceedings in Communications*, Vol. 144, No. 5, pp. 1-6, 1997.

[100] M. J. Karol, M. G. Hluchyj, S. P. Morgan, "Input versus output queueing on a space-division packet switch", *IEEE Transactions on Communications*, Vol. 35, No. 12, pp. 13471356, 1987.

[101] T. P. Troudet, S. M. Walters, "Neural network architecture for crossbar switch control", *IEEE Transactions on Circuits and Systems*, Vol. 38, No. 1, pp. 42-56, 1991.

[102] A. Varma, R. Antonucci, "A neural-network controller for scheduling packet transmissions in a crossbar switch", *Proceedings of the International Workshop on Applications of Neural Networks to Telecommunications* 2, pp. 121-128, 1995.

[103] T. X. Brown, "Neural networks for switching", *Neural Networks in Telecommunications, B.* B. Yuhas and N. Ansari editors, Boston: Kluwer Academic, pp. 11-36, 1994.

[104] H. E. Rauch, T. Winarske, "Neural networks for routing communication traffic", *IEEE Control System Magazine*, pp. 26-30, April 1988.

[105] L-D. Chou, J-L. C. Wu, "Buffer management using genetic algorithms and neural networks", *Proceedings of IEEE Globecom '95*, pp. 1333-1337, 1995.

[106] D. E. Goldberg, *Genetic Algorithms in Search, Optimisation and Machine Learning*, Addison-Wesley, 1989.

[107] B. Kosko, *Neural Networks and Fuzzy Systems : A Dynamical Systems Approch to Machine Intelligence*, Prentice-Hall Publishers, 1992.

[108] R. G. Cheng, C. J. Chang, "Design of a fuzzy traffic controller for ATM networks", *IEEE/ACM Transactions on Networking*, Vol. 4, No. 3, pp. 460-469, 1996.

[109] A. Pitsillides, Y. A. Sekercioglu, G. Ramamurthy, "Effective control of traffic flow in ATM networks using fuzzy explicit rate marking (FERM)", *IEEE Journal on Selected Areas in Communications*, Vol. 15, No. 2, pp. 209-225, 1997.

[110] K. Uehara, K. Hirota, "Fuzzy connection admission control for ATM networks based on possibility distribution of cell loss ratio", *IEEE Journal on Selected Areas in Communications*, Vol. 15, No. 2, pp. 179-190, 1997.

[111] K. F. Cheung et al. "Fuzzy logic based ATM policing", *Proceedings of ICCS '94*, pp. 535-539, 1994.

[112] V. Catania, G. Ficili, S. Palazzo, D. Panno, "Using fuzzy logic in ATM source traffic control : Lessons and perspectives", *IEEE Communications Magazine*, pp. 70-81, November 1996.

[113] T. D. Ndousse, "Fuzzy neural control of voice cells in ATM networks", *IEEE Journal on*

Selected Areas in Communications, Vol. 12, No. 9, pp. 1488-1494, 1994.

[114] A. R. Bonde, S. Ghosh, "A comparative study of fuzzy versus "fixed" thresholds for robust queue management in cell-switching networks", *IEEE/ACM Transactions on Networking*, Vol. 2, No. 4, pp. 337-344, 1994.

[115] K. A. Narendra, M. A. L. Thathachar, *Learning Automata : An Introduction*, Prentice-Hall Publishers, 1989.

[116] P. Mars, J. R. Chen, R. Nambiar, *Learning Algorithms : Theory and applications in signal processing, control and communications*, CRC Press Publishers, 1996.

[117] M. S. Chrystall, *Adaptive Control of Communication Networks using Learning Automata*, PhD Thesis, Robert Gordon Institute of Technology, Aberdeen, 1982.

[118] J. M. Reeve, *Learning Algorithms for the Control of Routing in Integrated Service Communication Networks*, PhD Thesis, University of Durham, 1998.

[119] L. G. Mason, X. Gu, "Learning automata models for adaptive flow control in packet-switching networks", *Proceedings of the Fourth Yale Workshop on Applications of Adaptive Systems Theory*, pp. 113-119, 1985.

[120] M. R. Meybodi, S. Lakshmivarahan, "A learning approach to priority assignment in a two class M/M/1 queueing system with unknown parameters", *Proceedings of the Third Yale Workshop on Applications of Adaptive Systems Theory*, pp. 106-109, 1983.

[121] H. J. Fowler, W. E. Leland, "Local area network traffic characteristics, with implications for broadband network congestion management", *IEEE Journal on Selected Areas in Communications*, Vol. 9, No. 7, 1991.

[122] M. C. Yuang, P. L. Tien, "Intelligent video smoother for multimedia communications", *Proceedings of IEEE Globecom '96*, pp. 502-507, 1996.

[123] W. M. Moh, M-J Chen, N-M. Chu, C-D Liao, "Traffic prediction and dynamic bandwidth allocation over ATM : A neural network approach", *Computer Communications*, Vol. 18, No. 8, pp. 563-571, 1995.

[124] K. Hornik, "Multilayer feedforward networks are universal approximators", *Neural Networks*, Vol. 2, pp. 359-366, 1989.

[125] A. A. Tarraf, I. W. Habib, T. N. Saadawi, "Neural networks for ATM multimedia traffic prediction", *Proceedings of the International Workshop on Applications of Neural Networks to Telecommunications*, pp. 85-91, 1993.

[126] E. S. Yu, C. Y. R. Chen, "Traffic prediction using neural networks", *Proceedings of IEEE Globecom '93*, pp. 991-995, 1993.

[127] T. Edwards, D. S. W. Tansley, R. J. Frank, N. Davey, "Traffic trends analysis using

neural networks", *Proceedings of the International Workshop on Applications of Neural Networks to Telecommunications*, pp. 157-164, 1997.

[128] M. Bromirski, W. Lobejko, "Neural prediction of self-similar ATM traffic", *5th IFIP Workshop on Performance Modelling and Evaluation of ATM Networks*, 1997.

[129] R. Drossu, T. V. Lakshman, Z. Obradovic, C. Raghavendra, "Single and multiple frame video traffic prediction using neural network models", *Computer Networks Architectures and Applications,* Chapman & Hall, pp. 146-158, 1995.

[130] P-R. Chang, J-T. Hu, "Optimal nonlinear adaptive prediction and modelling of MPEG video in ATM networks using pipelined recurrent neural networks", *IEEE Journal on Selected Areas in Communications*, Vol. 15, No. 6, pp.1087-1100, 1997.

[131] M. Garrett, W. Willinger, "Analysis, Modelling and Generation of self-similar VBR video traffic", *Proceedings of ACM SIGCOMM '94*, pp.269-280, August 1994.

[132] Z-L. Zhang, J. Kurose, J. Salehi, D. Towsley, "Smoothing, statistical multiplexing and call admission control for stored video", *IEEE Journal on Selected Areas in Communications*, Vol. 15, No. 6, pp. 1148-1166, 1997.

[133] E. A. Wan, "Time series prediction by using a connectionist network with internal delay lines", In *Time Series Prediction : Forecasting the Future and Understanding the Past* (A. S. Weigend and N. A. Gershenfeld, eds.), pp. 195-217, Addison-Wesley, 1994.

[134] P. L. Tien, M. C. Yuang, "Intelligent voice smoother for VBR voice over ATM networks", *Proceedings of IEEE Infocom '98*, pp. 841-848, 1998.

[135] Available from *ftp-info3.informatik.uni-werzburg.de/pub/MPEG*

[136] O. Rose, "Statistical properties of MPEG video traffic and their impact on traffic modelling in ATM systems", *Proceedings of the 20th Annual Conference on Local Computer Networks*, pp. 397-406, 1995.

[137] Available from *ftp.bellcore.com/pub/vbr.videotrace/*

[138] Internet Traffic Archive WWW site, *http://ita.ee.lbl.gov/index.html*

[139] K. J. Lang, G. E. Hinton, "The development of the time-delay neural network architecture for speech recognition", Technical Report CMU-CS-88-152. Carnegie Mellon University, Pittsburgh, 1988.

[140] S. Haykin. *Neural Networks: A Comprehensive Foundation.* Macmillan College Publishing Company, 1994.

[141] E. A. Wan, "Time series prediction by using a connectionist network with internal delay lines", *Time Series Prediction*, Addison-Wesley Publishing Company, pp. 195-217, 1994.

[142] T. Yang, "An optimal service scheduling policy for packet networks with quality of service

guarantees", *Proceedings of IEEE Globecom '96*, pp. 1657-1663, 1996.

[143] T. Yang, D. Tsang, S. li, "Cell scheduling and bandwidth allocation for a class of VBR video connections", *IEEE Workshop on Visual Signal Processing and Communications*, pp. 95-101, 1994.

[144] T. Yang, D. Tsang, P. McCabe, "Cell scheduling and bandwidth allocation for heterogeneous VBR video conferencing traffic", *Proceedings of IEEE Globecom '95*, pp. 371-377, 1995.

[145] L. Kleinrock, *Queueing Systems Volume I : Theory*, pp. 249-250, Wiley-Interscience Publishers, 1975.

[146] J. M. Peha, A. Tobagi, "Cost-based scheduling and dropping algorithms to support integrated services", *IEEE Transactions on Communications*, No. 2, Vol. 44, pp. 192-201, 1996.

[147] S. Jamin, P. B. Danzig, S. Shenker, L. Zhang, "A measurement-based admission control algorithm for integrated service packet networks", *IEEE/ACM Transactions on Networking*, Vol. 5, No.1, pp.56-70, 1997.

[148] S. Jamin, S. J. Shenker, P. B. Danzig, "Comparison of measurement based admission control algorithms for controlled-load services", *Proceedings of INFOCOM '97*, pp. 973-980, 1997.

[149] Jingyu Qui, E. W. Knightly, "QoS control via robust envelope-based MBAC", *Proceedings of IEEE International Workshop on QoS*, pp. 62-64, 1998.

[150] S. Floyd, "Comments on measurement-based admission control for controlled-load services", Lawrence Berkeley Laboratory Technical Report, 1996.

[151] R. J. Gibbens, F. P. Kelly, P. B. Key, "A decision-theoretic approach to call admission control in ATM networks", *IEEE Journal on Selected Areas in Communications*, Vol. 13, No. 6, pp. 1101-1114, 1995.

[152] S. Crosby, I. Leslie, B. McGurk, J. T. Lewis, R. Russell, F. Toomey, "Statistical properties of a near-optimal measurement-based CAC algorithm", *Proceedings of IEEE ATM '97 Workshop*, pp. 103-112, June 1997.

[153] C. Casetti, J. Kurose, D. Towsley, "An adaptive algorithm for measurement-based admission control in integrated services packet networks", University of Massachusetts Technical Report TR 96-76, 1996.

[154] M. Grossglauser, D. Tse, "A framework for robust measurement-based admission control", *Proceedings of ACM SIGCOMM '97*, 1997.

[155] D. Kouvatsos, *Private Communication*, University of Bradford, 1998.

[156] B. McGurk, C. Walsh, "Investigation of the performance of a measurement-based connection admission control algorithm", *5th IFIP Workshop on Performance Modelling and Evaluation of ATM Networks,* pp. 67/1-67/10, 1997.